

# HTML5



# Weekly plan (HTML5, 1<sup>st</sup> semester 2022)

- **wk01** : Introduction to curriculum, current state of HTML5 & github
- **wk02** : Making HTML5 documents
- **wk03** : iFrame, Media, my 1<sup>st</sup> Homepage & Intro to Semantic tags
- **wk04** : **Semantic tags & Web forms**
- **wk05** : **CSS3 I. Basic & Box model**
- **wk06** : **CSS3 II. Advanced & Animation**
- **wk07** : DIY:x-mas-card , **Quiz-15, my 2<sup>nd</sup> Homepage**
- **wk08** : Mid-term Exam. **my 2<sup>nd</sup> Homepage**
- **wk09** : **JS I. Data types & operators**
- **wk10** : **JS II. if, loop & functions**
- **wk11** : **JS III. Core objects & array**
- **wk12** : **JS IV. DOM & BOM**
- **wk13** : **JS V.**
- **wk14** : **my final Homepage (hmn\_rpt03)**
- **wk15** : Final exam.

this 활용하여 과제 해결



여기 클릭하면 크기와 색 변경

GoBack\_btn!!

# My ID (HTML5, 1<sup>st</sup> semester 2022)

ID	성명
HM01	김정현
HM02	오세현
HM03	김기덕
HM04	강대진
HM05	김성우
HM06	김창연
HM07	김창욱
HM08	김태화
HM09	박세훈
HM10	박신영

ID	성명
HM11	박제홍
HM12	이승무
HM13	이승준
HM14	이재하
HM15	이준희
HM16	이현준
HM17	임태형
HM18	정동현
HM19	정희서
HM20	김동영
HM21	정희철
HM22	조동현

# wk11-실습 : 결과를 나의 github에 올리기

4

실습 결과를 github에 올립니다.

1. README.md에는 실습 결과 요약 추가 입력
2. hmnn\_account\_prototype.html 완성
3. "hmxx" repo의 wk11 폴더에 upload

단 업로드가 안될 경우, wk11.zip을 업로드  
그리고 집에서 wk11 폴더로 다시 업로드.

# wk11-실습 : Account 사용자 객체 – prototype

5

```
<script>
  // 프로토타입 만들기 : 생성자 함수 작성
  function Account(owner, code, balance) {
    // 프로퍼티 만들기
    this.owner = owner; // 계좌 주인 프로퍼티 만들기
    this.code = code; // 계좌 코드 프로퍼티 만들기
    this.balance = balance; // 잔액 프로퍼티 만들기

    // 메소드 만들기
    this.inquiry = function () { return this.balance; }
    this.deposit = function (money) { this.balance += money; }
    this.withdraw = function (money) {
      this.balance -= money;
    }
  }
</script>
```

```
<script>
  var account = new Account("황기태", "111", 35000);

  document.write("account : ");
  document.write(account.owner + ", ");
  document.write(account.code + ", ");
  document.write(account.balance + "<br>");

  account.deposit(10000);
  document.write("10000원 저금 후 잔액은 " + account.inquiry() + "<br>");
  account.withdraw(5000);
  document.write("5000원 인출 후 잔액은 " + account.inquiry() + "<br>");

  document.write("<br>");

  var hm06 = new Account("김창연", "1238", 9876543210);

  document.write("account : ");
  document.write(hm06.owner + ", ");
  document.write(hm06.code + ", ");
  document.write(hm06.balance + "원<br>");

  hm06.deposit(123456790);
  document.write("123,456,790원 저금 후 잔액은 " + hm06.inquiry() + "원 입니다.<br>");
  hm06.withdraw(10000000000);
  document.write("1,000,000,000원 인출 후 잔액은 " + hm06.inquiry() + "원 입니다.<br>");

</script>
```



A vibrant field of sunflowers with bright yellow petals and dark brown centers, set against a clear blue sky with scattered white clouds. The sunflowers are in various stages of bloom, and their green leaves are visible at the base.

# 08

HTML DOM과 Document

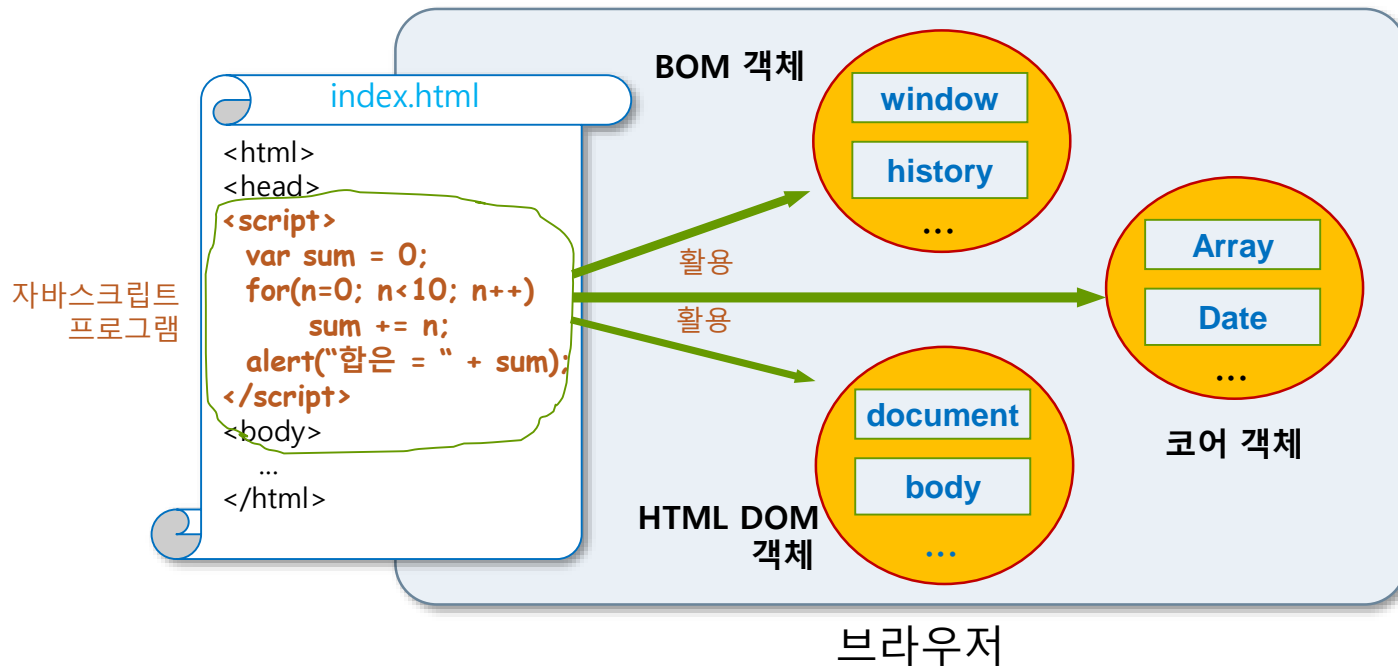
# 강의 목표

1. **HTML DOM**의 필요성을 이해한다.
2. **DOM 트리**와 HTML 페이지의 관계를 이해한다.
3. **DOM 객체**의 구조와 HTML 태그와의 관계를 이해한다.
4. DOM 객체를 통해 HTML 태그의 출력 모양과 콘텐츠를 제어할 수 있다.
5. **document 객체**를 이해하고, `write()` 메소드를 활용할 수 있다.
6. `createElement()` 등을 통해 **동적으로 DOM 객체**를 웹 페이지에 추가,삭제할 수 있다.
7. DOM project - DIY

# HTML 페이지와 자바스크립트 객체

8

- 자바스크립트 코드는 브라우저로부터 3 가지 유형의 객체를 제공 받아 활용할 수 있다.





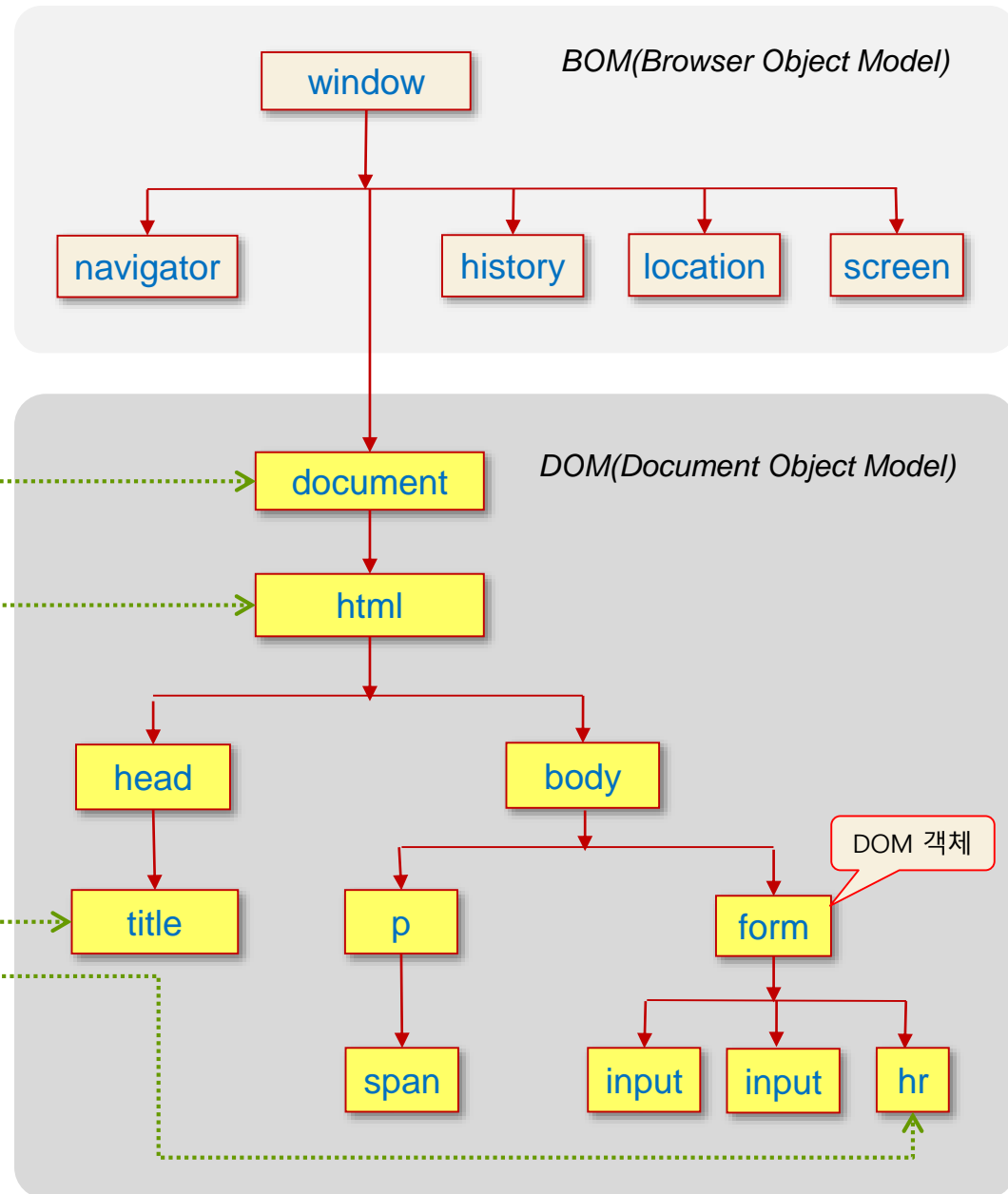
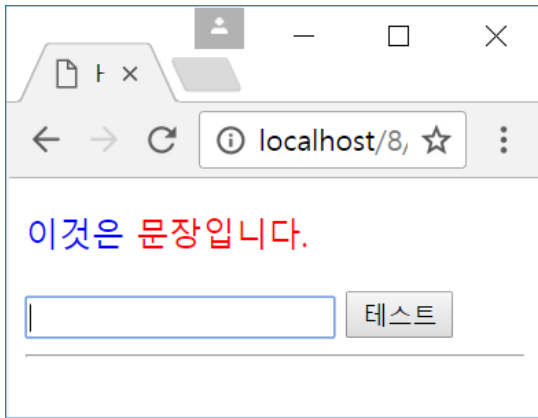
# HTML

# DOM

# HTML DOM (Document Object Model)

10

- HTML DOM(간단히 **DOM**)
  - ▣ 웹 페이지에 작성된 **HTML 태그** 당 객체(DOM 객체) 생성
  - ▣ 목적
    - HTML 태그가 출력된 모양이나 콘텐츠를 제어하기 위해
      - DOM 객체를 통해 각 태그의 *CSS3* 스타일 시트 접근 및 변경
      - HTML 태그에 의해 출력된 텍스트나 이미지 변경
- **DOM 트리**
  - ▣ HTML 태그의 포함관계에 따라 DOM 객체의 트리(tree) 생성
  - ▣ DOM 트리는 부모 자식 관계
- **DOM 객체**
  - ▣ DOM 트리의 한 노드(node)
  - ▣ HTML 태그 당 하나의 **DOM 객체** 생성
    - DOM **노드**(Node), DOM **엘리먼트**(Element) 라고도 불림



```
<!DOCTYPE html>
<html>
<head>
  <title> HTML DOM 트리 </title>
</head>
<body>
<p style="color:blue">이것은
  <span style="color:red">문장입니다.
  </span>
</p>
<form>
  <input type="text">
  <input type="button" value="테스트">
  <hr>
</form>
</body>
</html>
```

# DOM 트리 (tree)의 특징

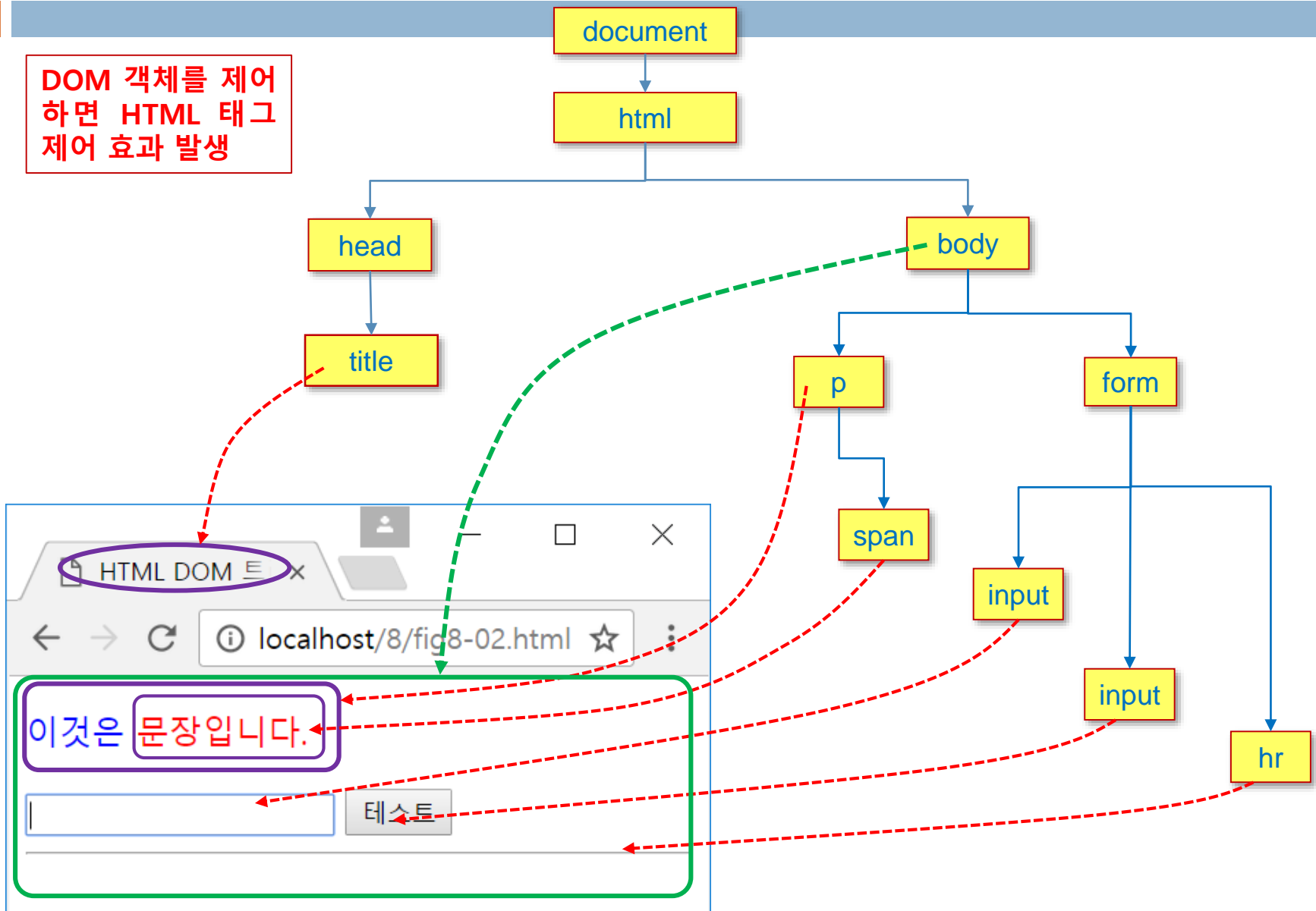
12

- DOM 트리의 특징
  - ▣ **DOM 트리의 루트는 document 객체**
  - ▣ DOM 객체의 종류는 HTML 태그 종류만큼 정의된다.
  - ▣ **HTML 태그 당 DOM 객체가 하나씩 생성**
  - ▣ **HTML 태그의 포함관계에 따라 DOM 트리에 부모 자식 관계**
- 브라우저가 HTML 태그를 화면에 그리는 과정
  1. 브라우저가 DOM 트리의 틀(document 객체) 생성
  2. 브라우저가 HTML 태그를 읽고 DOM 트리에 DOM 객체 생성
  3. 브라우저는 DOM 객체를 화면에 출력
  4. HTML 문서 로딩이 완료되면 DOM 트리 완성
  5. DOM 객체 변경 시, 브라우저는 해당 HTML 태그의 출력 모양을 바로 갱신

# DOM 객체와 HTML 페이지의 화면 출력

13

DOM 객체를 제어  
하면 HTML 태그  
제어 효과 발생



# HTML 태그의 요소

14

- HTML 태그
  - ▣ 엘리먼트(element)로도 불림
  - ▣ 다음 5 가지 요소로 구성
    - 엘리먼트 (태그) 이름
    - 속성
    - CSS3 스타일
    - 이벤트 리스너
    - 콘텐츠 (innerHTML)

태그이름  
(엘리먼트)

속성

CSS3 스타일

이벤트 리스너

```
<p id="firstP" style="color:blue" onclick="this.style.color='teal'">  
  이것은<span style="color:red">문장입니다.</span>  
</p>
```

콘텐츠(innerHTML)

A diagram illustrating the components of an HTML tag. The tag `<p id="firstP" style="color:blue" onclick="this.style.color='teal'">이것은<span style="color:red">문장입니다.</span></p>` is shown. Annotations with arrows point to different parts: '태그이름 (엘리먼트)' points to `<p>`, '속성' points to `id="firstP"`, 'CSS3 스타일' points to `style="color:blue"`, '이벤트 리스너' points to `onclick="this.style.color='teal'"`, and '콘텐츠(innerHTML)' points to the text `이것은<span style="color:red">문장입니다.</span>` inside the tag. The closing tag `</p>` is also present.



# DOM 객체의 구성 요소

15

- DOM 객체는 5 개의 요소 구성
  - ▣ 프로퍼티(property)
    - HTML 태그의 속성(attribute) 반영
  - ▣ 메소드(method)
    - DOM 객체의 멤버 함수로서, HTML 태그 제어 가능
  - ▣ 컬렉션(collection)
    - 자식 DOM 객체들의 주소를 가지는 등 배열과 비슷한 집합적 정보
  - ▣ 이벤트 리스너(event listener)
    - HTML 태그에 작성된 이벤트 리스너 반영
    - 약 70여개의 이벤트 리스너를 가질 수 있음
  - ▣ CSS3 스타일
    - HTML 태그에 설정된 CSS3 스타일 시트 정보를 반영
    - DOM 객체의 style 프로퍼티를 통해 HTML 태그의 모양 제어 가능

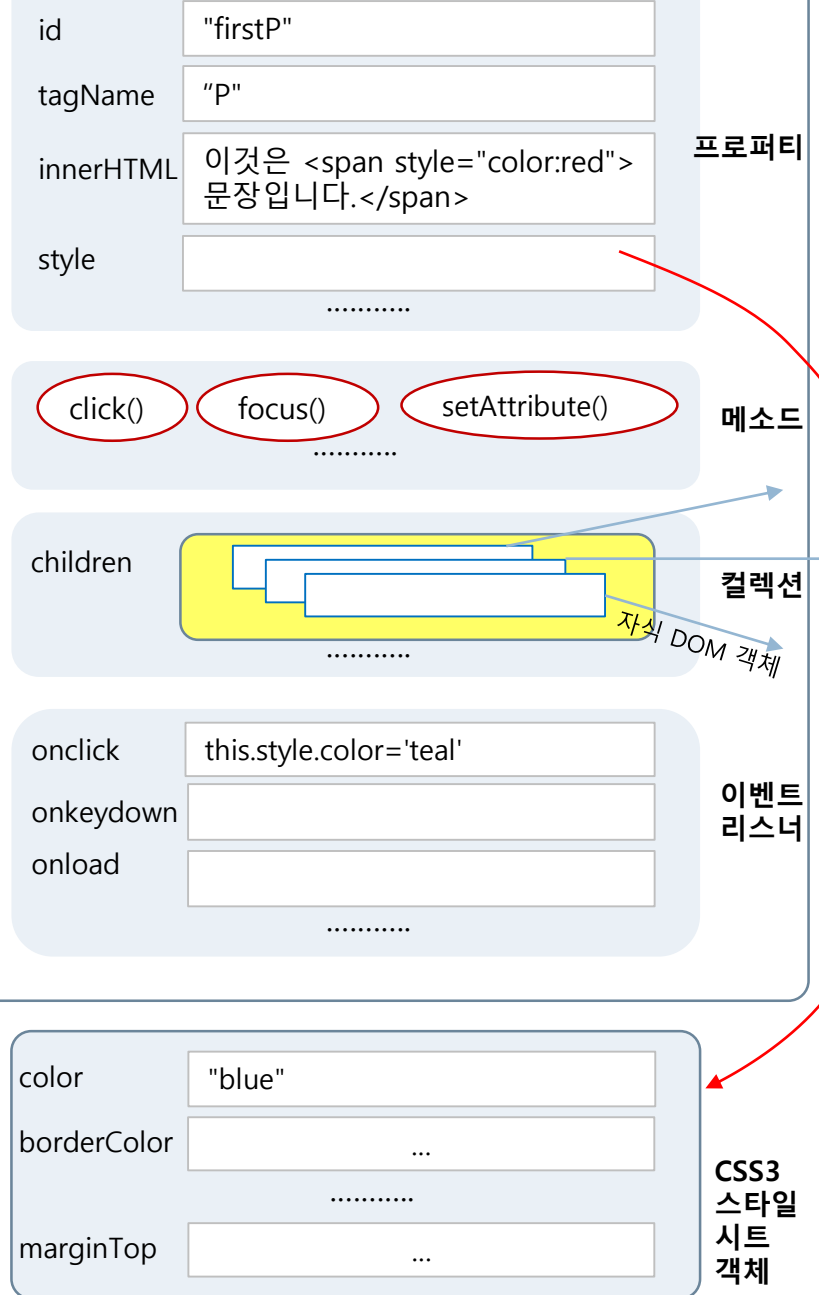
## DOM 객체의 구성

- 프로퍼티(property)
- 메소드(method)
- 컬렉션(collection)
- 이벤트 리스너(event listener)
- CSS3 스타일

```
<p id="firstP"  
  style="color:blue"  
  onclick="this.style.color='teal'">  
  이것은  
  <span style="color:red">  
    문장입니다.  
  </span>  
</p>
```

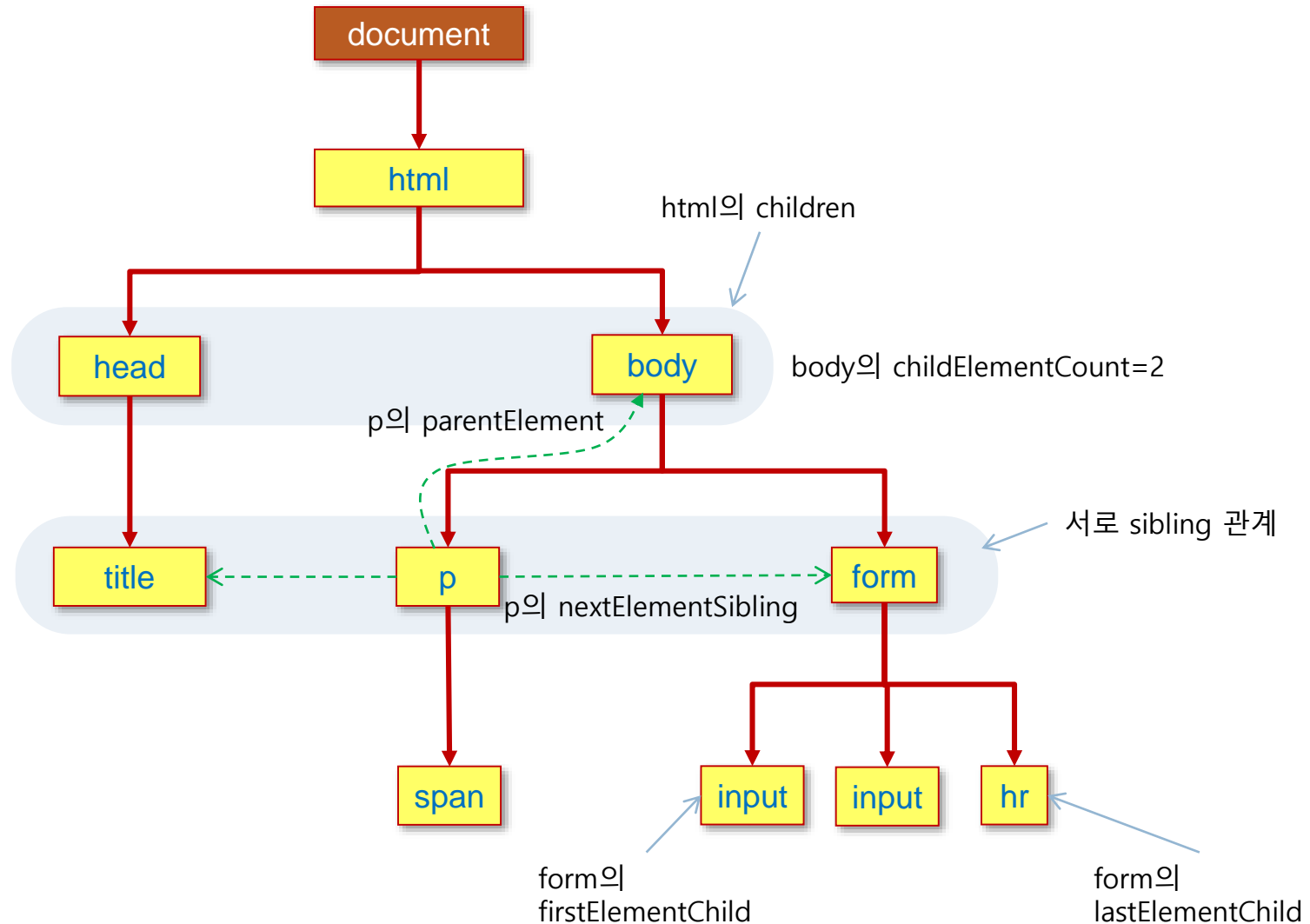
<p>...</p> 태그

## DOM 객체 p



# DOM 객체의 프로퍼티와 DOM 객체 사이의 관계

17



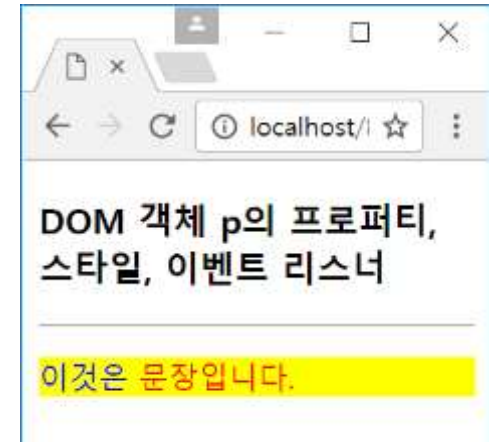
# 예제 8-1 DOM 객체의 구조 출력 : p 객체 사례

18

```
<!DOCTYPE html>
<html>
<head> <title>HTML DOM 트리</title> </head>
<body>
<h3>DOM 객체 p의 프로퍼티, 스타일, 이벤트 리스너</h3>
<hr>
<p id="firstP"
  style="color:blue; background:yellow"
  onclick="this.style.color='teal'">
  이것은 <span style="color:red">문장입니다.
</p>
<script>
  var p = document.getElementById("firstP");
  var text = "p.id = " + p.id + "\n";
  text += "p.tagName = " + p.tagName + "\n";
  text += "p.innerHTML = " + p.innerHTML + "\n";
  text += "p.style.color = " + p.style.color + "\n";
  text += "p.onclick = " + p.onclick + "\n";
  text += "p.childElementCount = " + p.childElementCount + "\n";
  text += "너비 = " + p.offsetWidth + "\n";
  text += "높이 = " + p.offsetHeight + "\n";
  alert(text);
</script>
</body>
</html>
```

id가 firstP인 태그의 DOM 찾기

wk12js\_ex01\_DOM\_structure.html



localhost 내용:

```
p.id = firstP
p.tagName = P
p.innerHTML = 이것은
□<span style="color:red">문장입니다.</span>

p.style.color = blue
p.onclick = function onclick(event) {
  this.style.color='teal'
}
p.childElementCount = 1
너비 = 234
높이 = 21
```

확인

# 예제 8-1 DOM 객체의 구조 출력 : p 객체 사례

19

DOM 객체 p의 프로퍼티, 스타일, 이벤트 리스너

이것은 문장입니다.  
다른 문장입니다.

<p> 태그에 다음을 추가하여 결과를 확인: <br><span style="color:blue">다른 문장입니다.</span>

JavaScript



p.id = firstP  
p.tagName = P  
p.innerHTML =

니다. </span>  
장입니다. </span>

p.style.color = blue  
p.onclick = function onclick(event) {  
 this.style.color='teal'  
}  
p.childElementCount = 3  
너비 = 463  
높이 = 40

이것은 <span style="color:red">문장입  
<br><span style="color:blue">다른 문

확인

# DOM 객체 다루기

20

- DOM 객체 구분, id 태그 속성

```
<p id="firstP">안녕하세요</p>
```

- DOM 객체 찾기, **document.getElementById()**

```
var p = document.getElementById("firstP"); // id 값이 firstP인 DOM 객체 리턴  
p.style.color = "red"; // p 객체의 글자 색을 red로 변경
```

- DOM 객체의 CSS3 스타일 동적 변경

- ▣ CSS3 스타일 프로퍼티는 다음과 같이 사용

- background-color 스타일 프로퍼티 -> backgroundColor
- font-size 스타일 프로퍼티 -> fontSize

```
<span id="mySpan" style="color:red">문장입니다.</span>
```

```
var span = document.getElementById("mySpan"); // id가 mySpan인 객체 찾기
```

```
span.style.color = "green"; // '문장입니다'의 글자 색을 green으로 변경
```

```
span.style.fontSize = "30px"; // '문장입니다'의 폰트를 30px 크기로 변경
```

```
span.style.border = "3px dotted magenta"; // 3픽셀의 magenta 점선 테두리
```



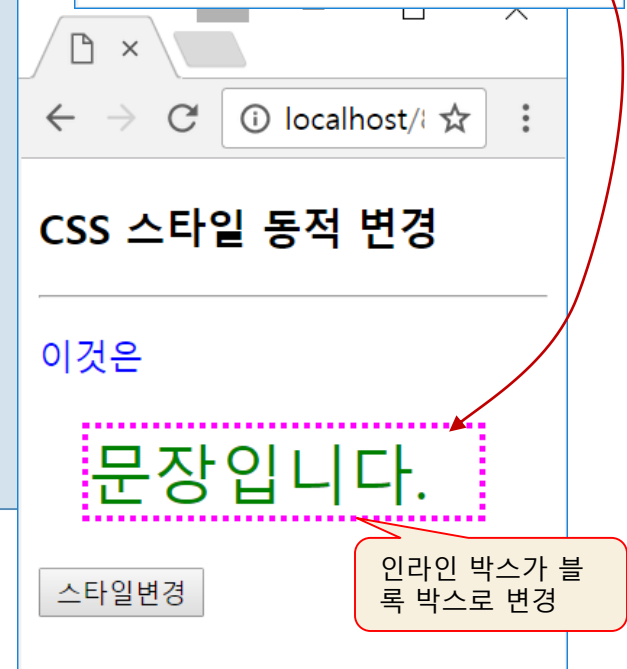
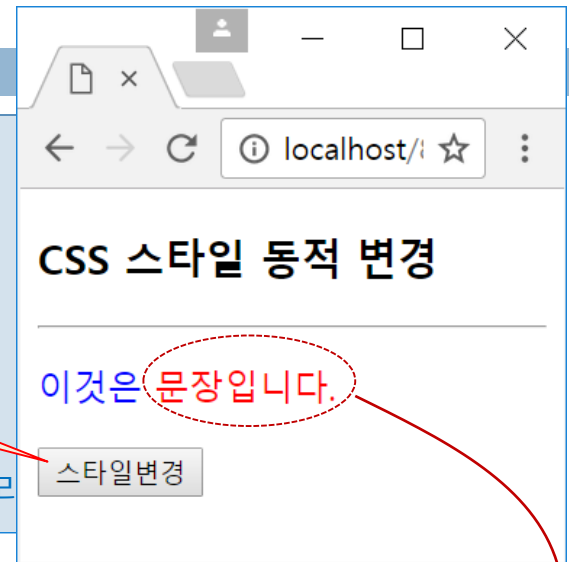
# 예제 8-2 <span>의 CSS3 스타일 동적 변경

21

```
<!DOCTYPE html>
<html><head><title>CSS 스타일 동적 변경</title>
<script>
function change() {
    var span = document.getElementById("mySpan");
    span.style.color = "green"; // 글자 색 green
    span.style.fontSize = "30px"; // 글자 크기는 30픽셀
    span.style.display = "block"; // 블록 박스로 변경
    span.style.width = "6em"; // 박스의 폭. 6 글자 크기
    span.style.border = "3px dotted magenta"; // 3픽셀 점선 magenta 테두리
    span.style.margin = "20px"; // 상하좌우 여백 20px
    span.style.padding = "10px";
}
</script>
</head>
<body>
<h3>CSS 스타일 동적 변경</h3>
<hr>
<p style="color:blue" >이것은
    <span id="mySpan" style="color:red">문장입니다.</span>
</p>
<input type="button" value="스타일변경" onclick="change()">
</body>
</html>
```

wk12js\_ex02\_DOM\_style.html

버튼을 클릭하면  
change() 함수 호출.  
스타일 변경



인라인 박스가 블  
록 박스로 변경

# innerHTML 프로퍼티

22

## □ innerHTML 프로퍼티

- ▣ 시작 태그와 종료 태그 사이에 들어 있는 HTML 콘텐츠

```
<p id="firstP" style="color:blue">  
  여기에 <span style="color:red">  
    클릭하세요.</span>  
</p>
```

innerHTML

- ▣ innerHTML 프로퍼티 수정 -> HTML 태그의 콘텐츠 변경

```
var p = document.getElementById("firstP");  
p.innerHTML= "나의 <img src='puppy.jpg'>강아지입니다.";
```



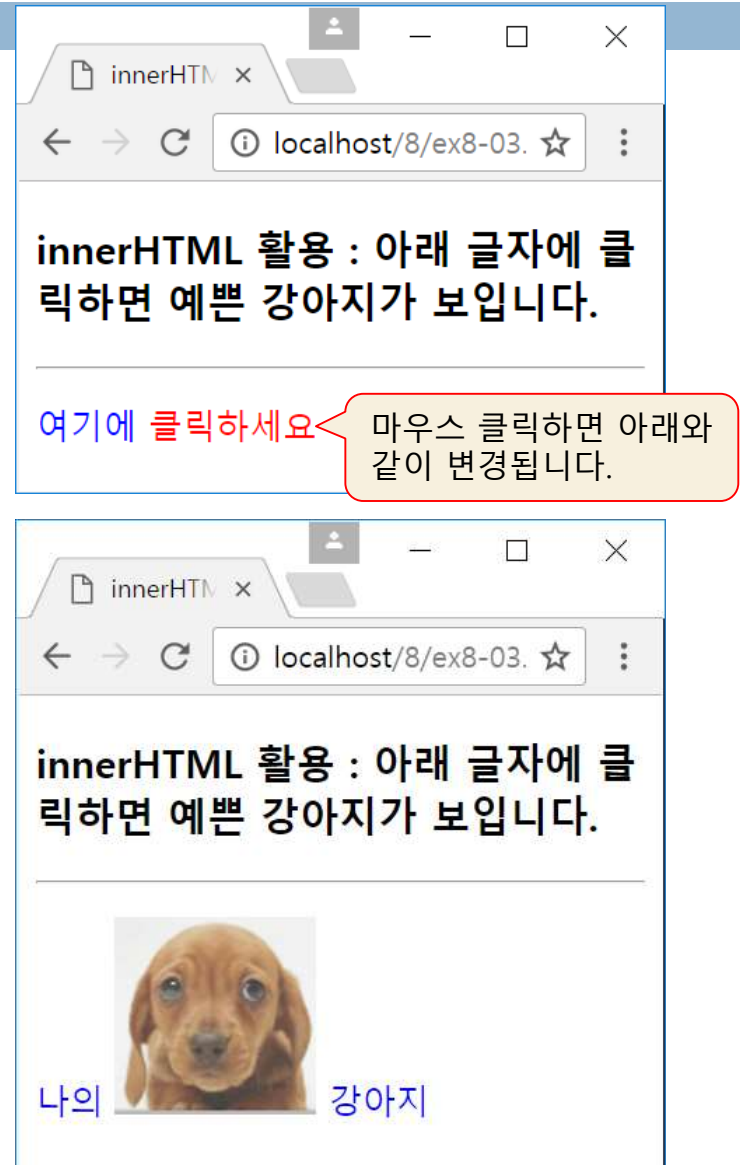
```
<p id="firstP" style="color:blue">  
  나의 <img src='puppy.jpg'>  
    강아지입니다.  
</p>
```

# 예제 8-3 innerHTML을 이용하여 HTML 콘텐츠 동적 변경

23

```
<!DOCTYPE html>
<html>
<head>
<title>innerHTML 활용</title>
<script>
function change() {
    var p = document.getElementById("firstP");
    p.innerHTML= "나의 <img src='media/puppy.png'>
강아지";
}
</script>
</head>
<body>
<h3>innerHTML 활용 : 아래 글자에 클릭하면
예쁜 강아지가 보입니다.</h3>
<hr>
<p id="firstP" style="color:blue"
    onclick="change()">
    여기에 <span style="color:red">클릭하세요</span>
</p>
</body>
</html>
```

wk12js\_ex03\_DOM\_innerHTML.html



## □ this 키워드

- ▣ 객체 자신을 가리키는 자바스크립트 키워드
- ▣ **DOM 객체에서 객체 자신을 가리키는 용도**로 사용
  - 예) <div> 태그 자신의 배경을 orange 색으로 변경

```
<div onclick="this.style.backgroundColor='orange'">
```

- 예) 버튼이 클릭되면 자신의 배경색을 orange로 변경

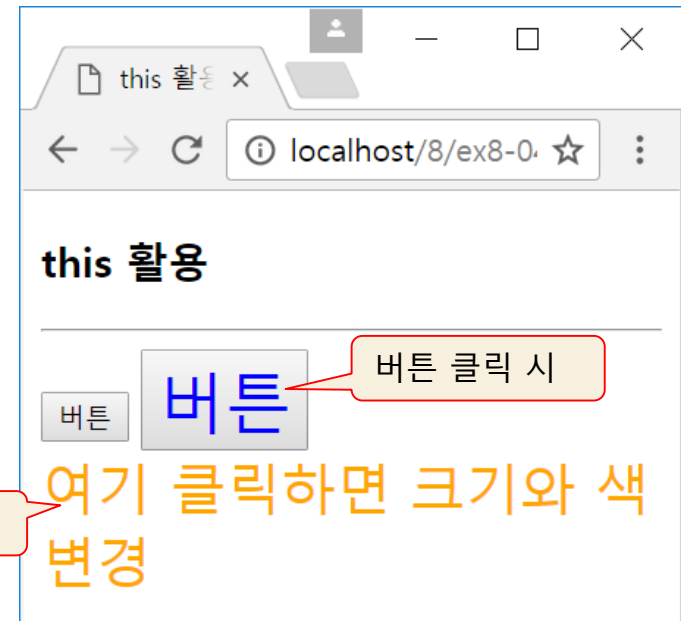
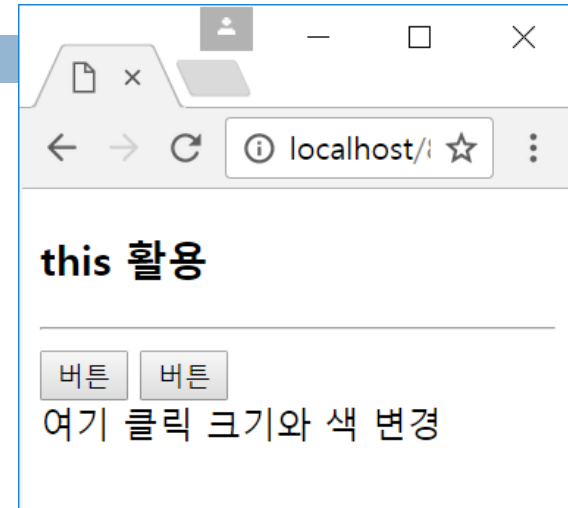
```
<button onclick="this.style.backgroundColor='orange'">
```

# 예제 8-4 this 활용 [DIY 과제]

25

```
<!DOCTYPE html>
<html>
<head>
<title>this 활용</title>
<script>
function change(obj, size, color) {
  obj.style.color = color;
  obj.style.fontSize = size;
}
</script>
</head>
<body>
<h3>this 활용</h3>
<hr>
<button onclick="change(this, '30px', 'red')">버튼</button>
<button onclick="change(this, '30px', 'blue')">버튼</button>
<div onclick="change(this, '25px', 'orange')">
  여기 클릭하면 크기와 색 변경
</div>
</body>
</html>
```

wk12js\_ex04\_DOM\_this.html



텍스트 클릭 시

# HTML

# document 객체



# document 객체

27

## □ document

### ▣ HTML 문서 전체를 대변하는 객체

- 프로퍼티 - HTML 문서의 전반적인 속성 내포
- 메소드 - DOM 객체 검색, DOM 객체 생성, HTML 문서 전반적 제어

### ▣ DOM 객체를 접근하는 경로의 시작점

### ▣ DOM 트리의 최상위 객체

- 브라우저는 HTML 문서 로드 전, document 객체를 먼저 생성
- document 객체를 뿌리(root)로 하여 DOM 트리 생성

## □ document 객체 접근

### ▣ window.document 또는 document 이름으로 접근

### ▣ document 객체는 DOM 객체가 아님

- 연결된 스타일 시트가 없음

```
document.style.color = "red"; // 오류. document에는 CSS3 스타일 시트가 연결되지 않음
```

# 예제 8-5 document 객체의 프로퍼티 출력

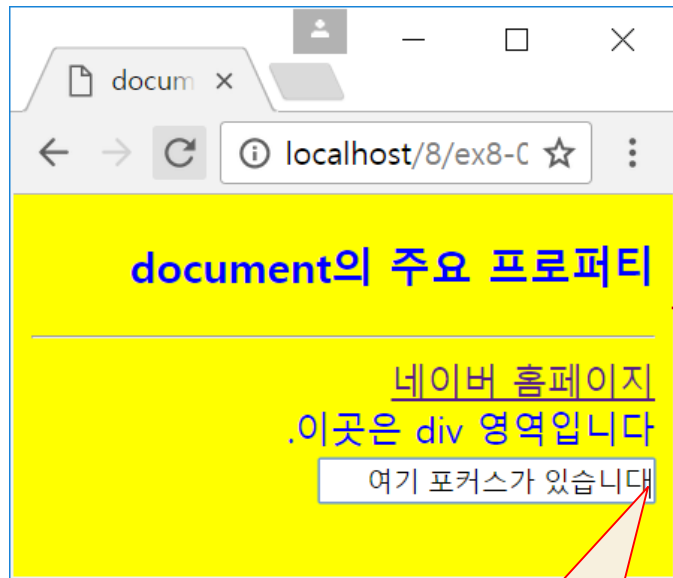
28

```
<!DOCTYPE html>
<html>
<head id="myHead">
<title>document 객체의 주요 프로퍼티</title>
<script>
    var text = "문서 로딩 중일 때 readyState = " + document.readyState + "
</script>
</head>
<body style="background-color:yellow; color:blue; direction:rtl" onload="printProperties()">
<h3>document의 주요 프로퍼티</h3>
<hr>
<a href="http://www.naver.com">네이버 홈페이지</a>
<div>이곳은 div 영역입니다.</div>
<input id="input" type="text" value="여기 포커스가 있습니다">
<script>
// 문서가 완전히 로드(출력)되었을 때, 현재 document의 프로퍼티 출력
function printProperties() {
    document.getElementById("input").focus(); // <input> 태그에 포커스를 줌

    text += "1. location = " + document.location + "
    text += "2. URL = " + document.URL + "
    text += "3. title = " + document.title + "
    text += "4. head의 id = " + document.head.id + "
    text += "5. body color = " + document.body.style.color + "
    text += "6. domain = " + document.domain + "
    text += "7. lastModified = " + document.lastModified + "
    text += "8. defaultView = " + document.defaultView + "
    text += "9. 문서의 로드 완료 후 readyState = " + document.readyState + "
    text += "10. referrer = " + document.referrer + "
    text += "11. activeElement = " + document.activeElement.value + "
    text += "12. documentElement의 태그 이름 = " + document.documentElement.tagName + "
    alert(text);
}
</script>
</body>
</html>
```

# 예제 8-5 document 객체의 프로퍼티 출력

29



로드 후  
경고창  
출력

localhost 내용:

- 문서 로딩 중일 때 readyState = loading
- 1. location =http://localhost/8/ex8-05.html
- 2. URL =http://localhost/8/ex8-05.html
- 3. title =document 객체의 주요 프로퍼티
- 4. head의 id =myHead
- 5. body color =blue
- 6. domain =localhost
- 7. lastModified =10/28/2016 09:47:56
- 8. defaultView = [object Window]
- 9. 문서의 로드 완료 후 readyState = complete
- 10. referrer =
- 11. activeElement = 여기 포커스가 있습니다
- 12. documentElement의 태그 이름 = HTML

확인

경고창에 document 객체의 주요 프로퍼티 출력

# DOM 트리에서 DOM 객체 찾기

30

## □ 태그 이름으로 찾기

### ▣ `document.getElementsByTagName()`

- 태그 이름이 같은 모든 DOM 객체들을 찾아 컬렉션 리턴
- 예) <div> 태그의 모든 DOM 객체 찾기

```
var divTags = document.getElementsByTagName("div");
```

```
var n = divTags.length; // 웹 페이지에 있는 <div> 태그의 개수
```

## □ class 속성으로 찾기

### ▣ `document.getElementsByClassName()`

- class 속성이 같은 모든 DOM 객체들을 찾아 컬렉션 리턴
- 예)

```
<div class="plain">...</div>  
<div class="important">...</div>  
<div class="plain">...</div>
```

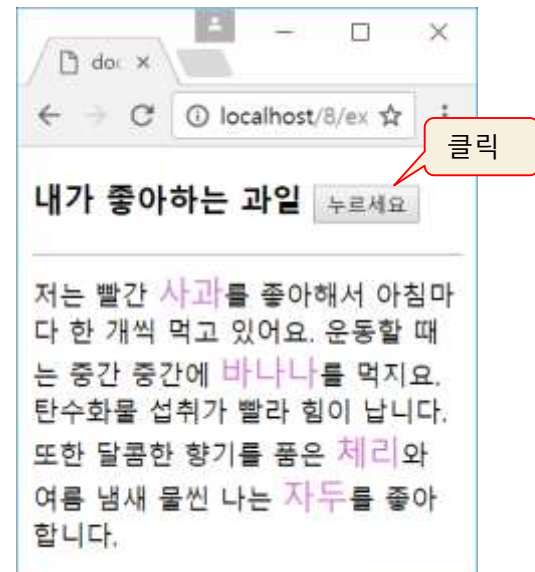
```
var plainClasses = document.getElementsByClassName("plain");  
var n = plainClasses.length; // 웹 페이지에 class="plain" 속성을 가진 태그의 개수
```

# 예제 8-6 태그 이름으로 DOM 객체 찾기, **getElementsByTagName()**

31

```
<!DOCTYPE html>
<html>
<head>
<title>document.getElementsByTagName()</title>
<script>
function change() {
    var spanArray = document.getElementsByTagName("span");
    for(var i=0; i<spanArray.length; i++) {
        var span = spanArray[i];
        span.style.color = "orchid";
        span.style.fontSize = "20px";
    }
}
</script>
</head>
<body>
<h3>내가 좋아하는 과일
    <button onclick="change()">누르세요</button>
</h3>
<hr>
저는 빨간 <span>사과</span>를 좋아해서
아침마다 한 개씩 먹고 있어요. 운동할 때는 중간 중간에
<span>바나나</span>를 먹지요. 탄수화물 섭취가 빨라
힘이 납니다. 또한 달콤한 향기를 품은 <span>체리</span>와
여름 냄새 물씬 나는 <span>자두</span>를 좋아합니다.
</body>
</html>
```

wk12js\_ex06\_DOM\_byTag.html



# document.write()와 document.writeln()

32

## □ HTML 페이지 로딩 과정

1. 브라우저는 HTML 페이지 로드 전 빈 상태 document 생성
2. 브라우저는 HTML 페이지를 위에서 아래로 해석
3. HTML 태그들을 document 객체에 담아간다 (DOM 객체 생성).
4. </html> 태그를 만나면 document 객체를 완성하고 닫는다.

## □ write()

- document 객체에 담긴 HTML 콘텐츠 마지막에 HTML 태그들을 추가

- 추가되는 HTML 태그들은 DOM 객체로 바뀌고 DOM 트리에 추가
- 삽입된 HTML 태그들이 브라우저 화면에 출력
- 예)

```
document.write("<h3>Welcome to my home</h3>");  
document.write(2+3); // 합한 결과 5 출력  
document.write("<p>오늘은 " + "sunny day 입니다</p>");
```

## □ writeln()

- HTML 텍스트에 '\n'을 덧붙여 출력. **한 칸 띄는 효과**
- 한줄을 띄려면

```
document.write("<br>");
```



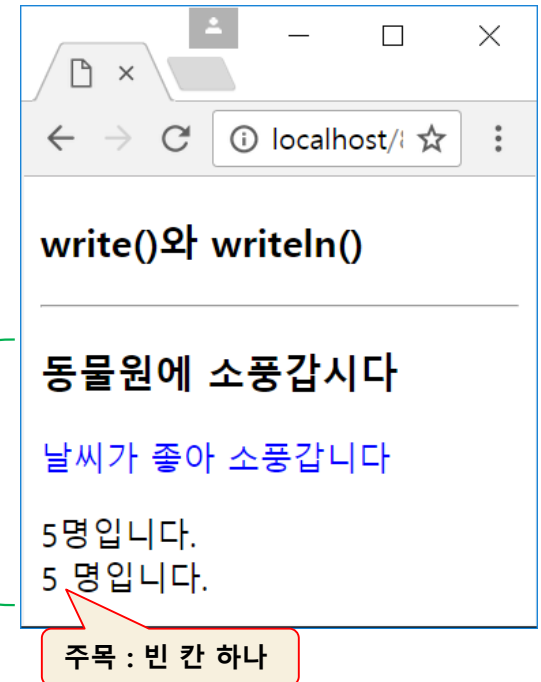
# 예제 8-7 write()와 writeln() 메소드 활용

33

```
<!DOCTYPE html>
<html>
<head>
<title>write()와 writeln() 활용</title>
</head>
<body>
<h3>write()와 writeln() 활용</h3>
<hr>
<script>
  document.write("<h3>동물원에 소풍갑시다</h3>");
  document.write("<p style='color:blue'>날씨가 좋아 ");
  document.write("소풍갑니다</p>");
  document.write(2+3);
  document.write("명입니다.<br>"); // 다음 줄로 넘어가기

  document.writeln(5); // 다음 줄에 넘어가지 못함
  document.write("명입니다.<br>");
</script>
</body>
</html>
```

wk12js\_ex07\_DOM\_write.html

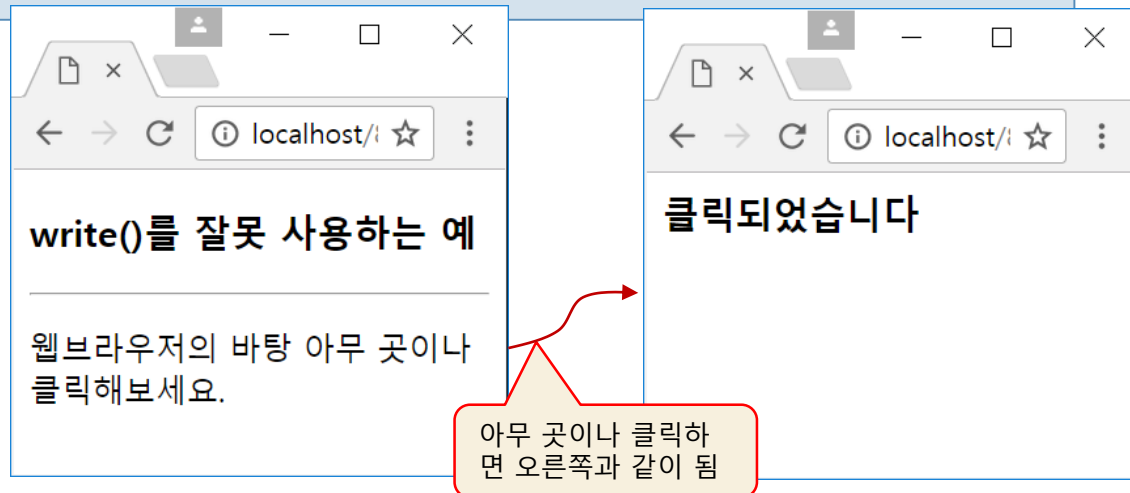


# 예제 8-8 write()를 잘못 사용하는 예

34

```
<!DOCTYPE html>
<html>
<head>
<title>write()를 잘못 사용하는 예</title>
</head>
<body onclick="document.write('<h3>클릭되었습니다</h3>')">
<h3>write()를 잘못 사용하는 예</h3>
<hr>
<p>웹브라우저의 바탕 아무 곳이나 클릭해보세요.</p>
</body>
</html>
```

wk12js\_ex08\_DOM\_write\_wrong.html



# document의 열기와 닫기, open()과 close()

35

## □ document.open()

- ▣ 현재 브라우저에 출력된 HTML 콘텐츠를 지우고 새로운 HTML 페이지 시작. 즉 document 객체에 담긴 DOM 트리를 지우고 새로 시작

## □ document.close()

- ▣ 현재 브라우저에 출력된 HTML 페이지 완성
- ▣ 더 이상 document.write() 할 수 없음

## □ 예)

```
// 현재 HTML 페이지의 내용을 지우고 다시 시작
```

```
document.open();  
document.write("<html><head>...<body>안녕하세요.");  
document.write(".....");  
document.write("</body> </html>");  
document.close();
```

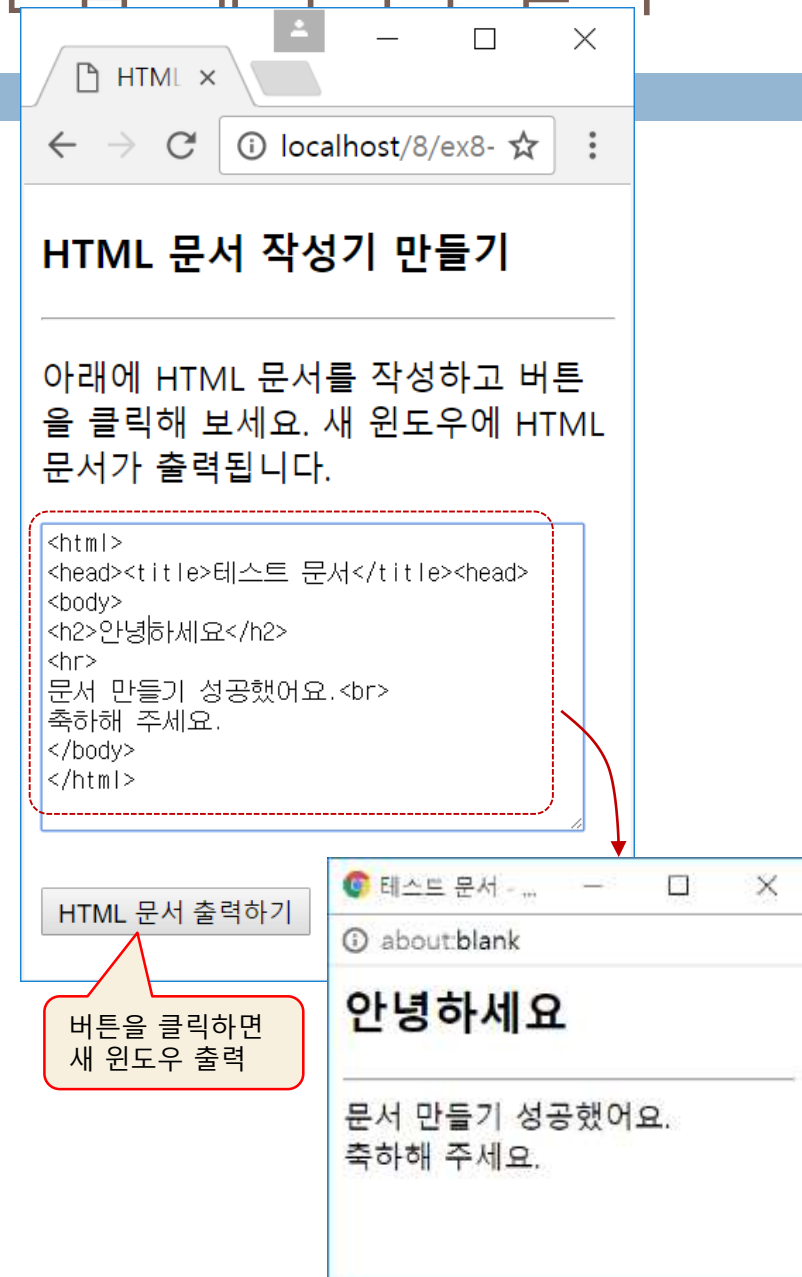
# 예제 8-9 HTML 문서 작성 연습 페이지 만들기

36

```
<!DOCTYPE html>
<html>
<head> <title>HTML 문서 작성기 만들기</title>
<script>
var win=null;
function showHTML() {
  if(win == null || win.closed)
    win = window.open("", "outWin", "width=300,height=200");

  var textArea = document.getElementById("srcText");
  win.document.open();
  win.document.write(textArea.value);
  win.document.close();
}
</script>
</head>
<body>
<h3>HTML 문서 작성기 만들기 </h3>
<hr>
<p>아래에 HTML 문서를 작성하고 버튼을 클릭해 보세요.
새 윈도우에 HTML 문서가 출력됩니다.</p>
<textarea id="srcText" rows="10" cols="50"> </textarea>
<br>
<br>
<button onclick="showHTML()">HTML 문서 출력하기</button>
</body>
</html>
```

wk12js\_ex09\_DOM\_html\_maker.html



# 예제 8-9 HTML 문서 작성 연습 페이지 만들기

37

## Syntax

```
window.open(URL, name, specs, replace)
```

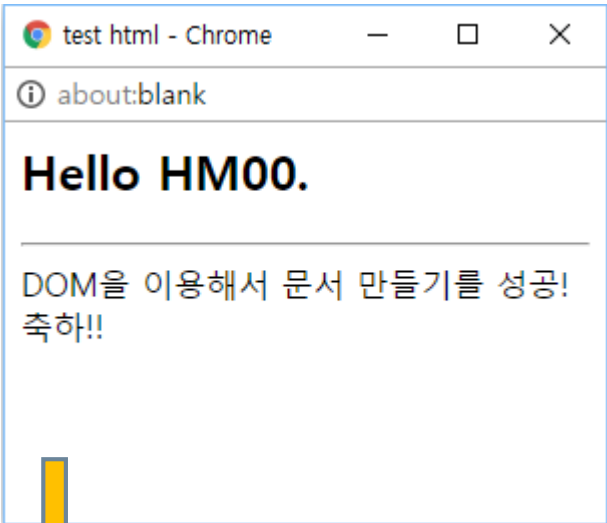
[https://www.w3schools.com/jsref/met\\_win\\_open.asp](https://www.w3schools.com/jsref/met_win_open.asp)

## HTML 문서 작성기 만들기

아래에 HTML 문서를 작성하고 버튼을 클릭해 보세요.  
새 윈도우에 HTML 문서가 출력됩니다.

```
<html>
<head><title>test html</title></head>
<body>
<h2>Hello HM00.</h2>
<hr>
DOM을 이용해서 문서 만들기를 성공! <br>
축하!!
</body>
</html>
```

HTML 문서 출력하기



**HMxx\_html\_maker.png**  
로 저장하고 github에 올림.

# 문서의 동적 구성

38

## □ DOM 객체 동적 생성: document.createElement("태그이름")

### □ 태그이름의 DOM 객체 생성

#### ■ 예)

```
var newDIV = document.createElement("div");
```

```
newDIV.innerHTML = "새로 생성된 DIV입니다.";
```

```
newDIV.setAttribute("id", "myDiv");  
newDIV.style.backgroundColor = "yellow";
```

## □ DOM 트리에 삽입

### □ 부모.appendChild(DOM객체);

### □ 부모.insertBefore(DOM객체 [, 기준자식]);

#### ■ 예) 생성한 <div> 태그를 <p id=p> 태그의 마지막 자식으로 추가

```
var p = document.getElementById("p");  
p.appendChild(newDiv);
```

## □ DOM 객체의 삭제

### □ var removedObj = 부모.removeChild(떼어내고자하는자식객체);

#### ■ 예)

```
var myDiv = document.getElementById("myDiv");  
var parent = myDiv.parentElement;  
parent.removeChild(myDiv); // 부모에서 myDiv 객체 삭제
```

# <div> 태그의 DOM 객체 동적 생성

39

```
var newDIV = document.createElement("div");  
newDIV.innerHTML = "새로 생성된 DIV입니다.";  
newDIV.setAttribute("id", "myDiv");  
newDIV.style.backgroundColor = "yellow";
```



```
<div id="myDiv"  
      style="background-color:yellow">  
  새로 생성된 DIV입니다.  
</div>
```

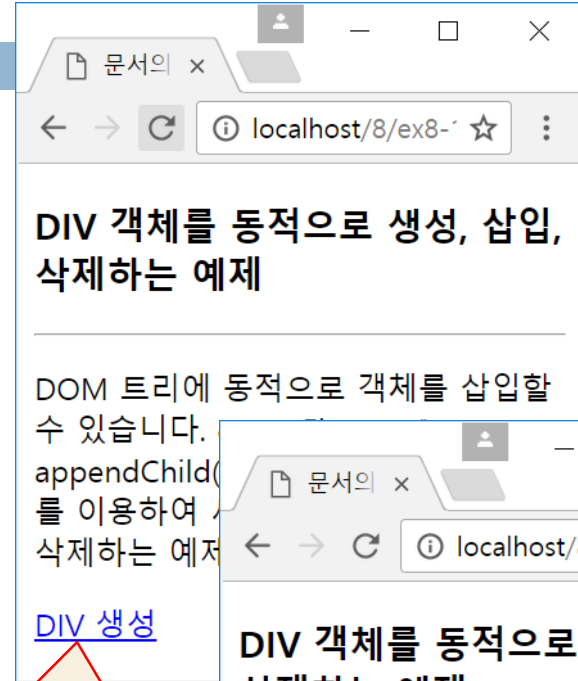
\* 이 자바스크립트 코드는 사실상 오른쪽의  
<div> 태그 정보를 가진 DOM 객체 생성

# 예제 8-10 HTML 태그의 동적 추가 및 삭제 (exam)

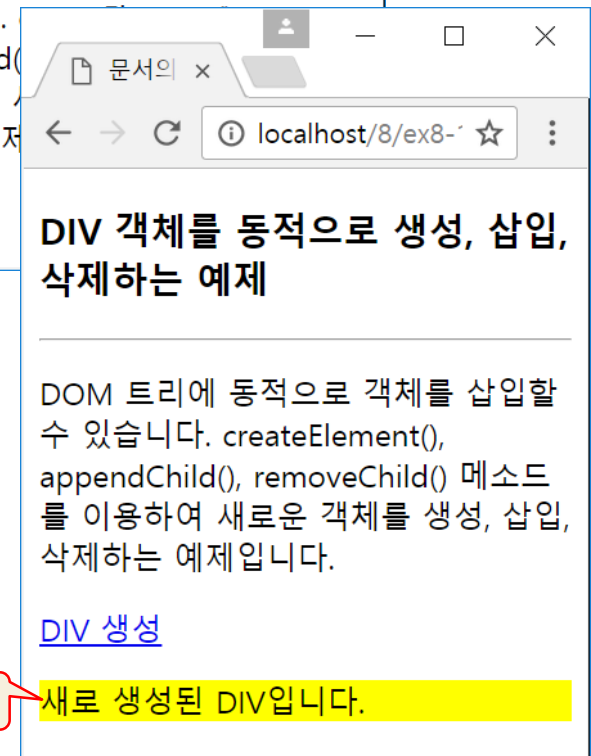
40

```
<!DOCTYPE html>
<html>
<head> <title>문서의 동적 구성</title>
<script>
function createDIV() {
    var obj = document.getElementById("parent");
    var newDIV = document.createElement("div");
    newDIV.innerHTML = "새로 동적 생성된 DIV입니다.";
    newDIV.setAttribute("id", "myDiv");
    newDIV.style.backgroundColor = "yellow";
    newDIV.onclick = function() {
        var p = this.parentElement; // 부모 HTML 태그 요소
        p.removeChild(this); // 자신을 부모로부터 제거
    };
    obj.appendChild(newDIV);
}
</script>
</head>
<body id="parent">
<h3>DIV 객체를 동적으로 생성, 삽입, 삭제하는 예제</h3>
<hr>
<p>DOM 트리에 동적으로 객체를 삽입할 수 있습니다.
createElement(), appendChild(),
removeChild() 메소드를 이용하여 새로운 객체를 생성,
삽입, 삭제하는 예제입니다.</p>
<a href="javascript:createDIV()">DIV 생성</a>
<p>
</body>
</html>
```

wk12js\_ex10\_DOM\_dynamic\_element.html



클릭하면 아래와 같이  
<div> 태그가 삽입



클릭하면 삭제





10

윈도우와 브라우저 관련 객체

# 강의 목표

1. **BOM**, 즉 브라우저 관련 객체 종류를 안다.
2. **window** 객체를 이해하고 **윈도우 열기, 닫기** 등을 제어할 수 있다.
3. **window** **객체의 타이머 기능**을 활용할 수 있다. (중요)
4. **window** 객체를 이용하여 **프린트, 윈도우 움직이기** 등 다양한 제어를 할 수 있다.
5. **location** 객체로 윈도우에 로드된 문서의 주소를 알고 새 문서를 로드할 수 있다.
6. **navigator** 객체를 통해 현재 브라우저의 관한 정보를 알아낼 수 있다.
7. **screen** 객체를 통해 현재 스크린 장치의 해상도를 알아 낼 수 있다.
8. **history** 객체를 이용하여 지금까지 윈도우에 로드된 웹 페이지로 이동할 수 있다.
9. BOM project - DIY

# HTML

# BOM

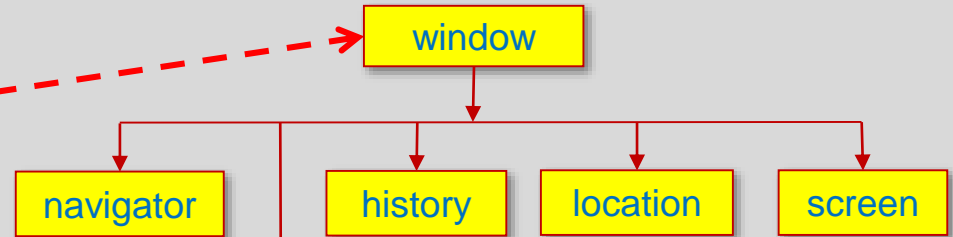
# 브라우저 관련 객체 개요

44

- **BOM(Browser Object Model) 객체들**
  - ▣ 자바스크립트로 브라우저를 제어하기 위해 지원되는 객체들
    - HTML 페이지의 내용과 관계없음
  - ▣ 브라우저 공통 BOM 객체들과 기능
    - **window** - 브라우저 윈도우 모양 제어. 새 윈도우 열기/닫기
    - **navigator** - 브라우저에 대한 다양한 정보 제공
    - **history** - 브라우저 윈도우에 로드한 URL 리스트의 히스토리 관리
    - **location** - 브라우저 윈도우에 로드된 HTML 페이지의 URL 관리
    - **screen** - 브라우저가 실행되고 있는 스크린 장치에 대한 정보 제공
- **BOM의 국제 표준이 없다.**
  - ▣ 브라우저마다 BOM 객체들이 조금씩 다름
  - ▣ 브라우저마다 이름이 같은 BOM 객체의 프로퍼티와 메소드 상이

브라우저 관련 객체들

*BOM(Browser Object Model)*



이것은 문장입니다.

document

*DOM(Document Object Model)*

html

head

body

title

p

form

span

input

input

hr

```
<!DOCTYPE html>
<html>
<head>
  <title>HTML DOM 트리</title>
</head>
<body>
  <p style="color:blue" >이것은
    <span style="color:red">문장입니다.
  </span>
</p>
  <form>
    <input type="text" name="s">
    <input type="button" value="테스트">
    <hr>
  </form>
</body>
</html>
```

HTML 문서의 내용과  
관련된 객체들

# BOM

# window 객체

# window 객체

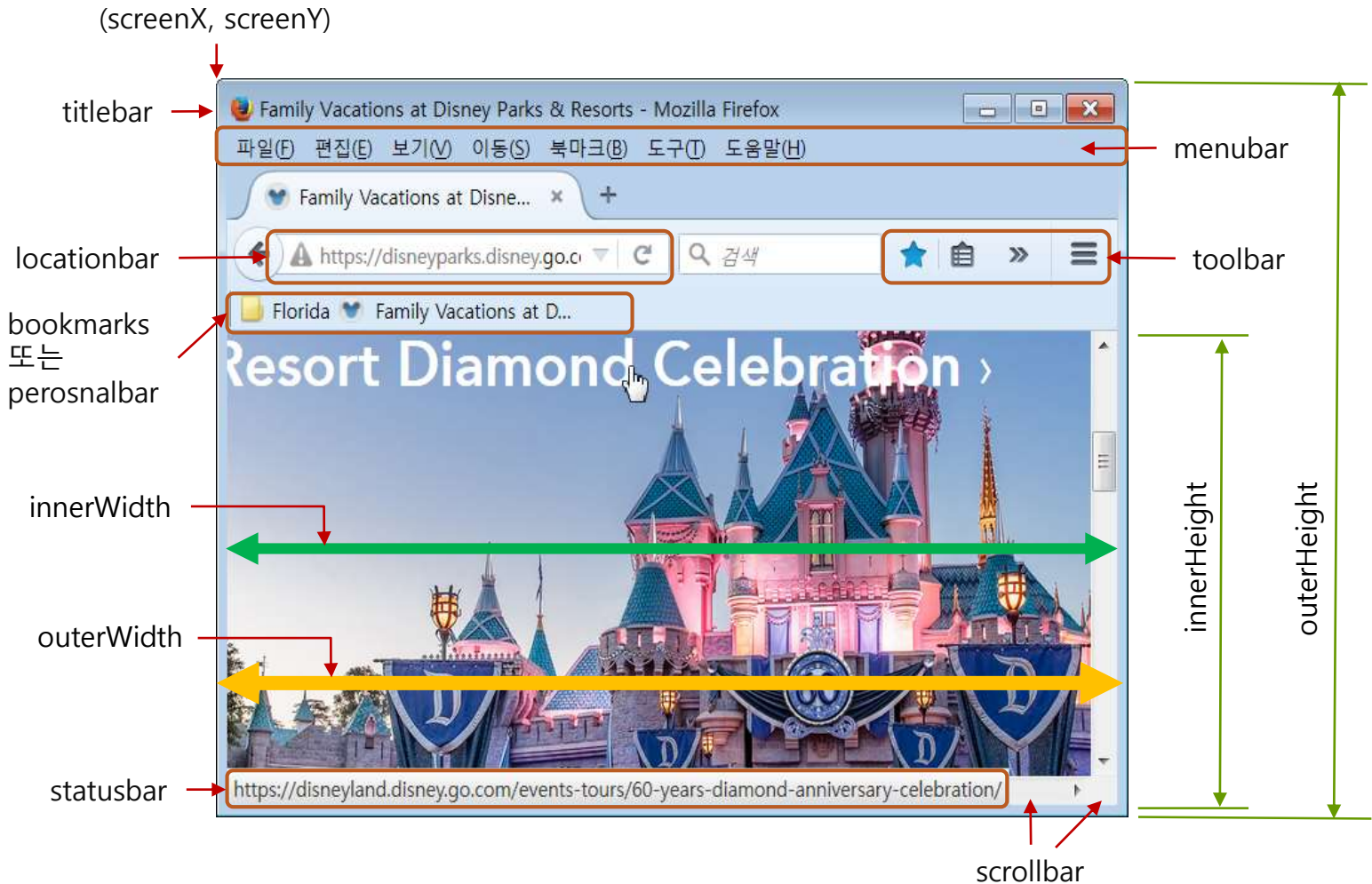
47

- window 객체
  - ▣ 열려 있는 브라우저 윈도우나 탭 윈도우의 속성을 나타내는 객체
  - ▣ 브라우저 윈도우나 탭 윈도우마다 별도의 window 객체 생성
  - ▣ **Html 문서가 담기는 틀**
- window 객체의 생성
  - ▣ 3 가지 경우
    - 브라우저가 새로운 웹 페이지를 로드할 때
    - <iframe> 태그 당 하나의 window 객체 생성
    - 자바스크립트 코드로 윈도우 열기 시 window 객체 생성
      - `window.open("웹페이지 URL", "윈도우이름", "윈도우속성")`,
- 자바스크립트 코드로 윈도우 객체에 대한 접근
  - ▣ **window**, 혹은 **window.self**, 혹은 **self**



# 윈도우 모양과 window 객체의 프로퍼티

48





# 윈도우 열기

49

## □ window.open()

- 윈도우를 새로 열고 웹 페이지 출력

■ 예)

```
window.open("http://www.naver.com", "", "");
```

- 3개의 매개변수를 가진 함수

```
window.open(sURL, sWindowName, sFeature)
```

- sURL : 윈도우에 출력할 웹 페이지 주소 문자열
- sWindowName : 새로 여는 윈도우의 이름 문자열로서 생략 가능
- sFeature : 윈도우의 모양, 크기 등의 속성들을 표현하는 문자열. 속성들은 빈칸 없이 콤마(‘,’)로 분리하여 작성하며 생략 가능

- 윈도우 이름(sWindowName)

_blank :	이름 없는 새 윈도우를 열고, 웹 페이지 로드
_parent :	현재 윈도우(혹은 프레임)의 부모 윈도우에 웹 페이지 로드
_self :	현재 윈도우에 웹 페이지 로드
_top :	브라우저 윈도우에 웹 페이지 로드

# 윈도우 열기 사례

50

- myWin 이름에 툴바만 가지는 새 윈도우 열고 sample.html 출력

```
window.open("sample.html", "myWin", "toolbar=yes");
```

- 현재 윈도우에 sample.html 출력

```
window.open("sample.html", "_self");
```

- 이름 없는 새 윈도우에 sample.html 출력

```
window.open("sample.html", "_blank");
```

- (10, 10) 위치에 300x400 크기의 새 윈도우 열고 네이버 페이지 출력

```
window.open("http://www.naver.com", "myWin",  
            "left=10,top=10,width=300,height=400");
```

- 이름과 속성이 없는 윈도우 열기

```
window.open("http://www.naver.com");  
window.open("http://www.naver.com", null, "");
```

- 빈 윈도우 생성

```
window.open();  
window.open("");
```

```
window.open("", "", "");  
window.open("", null, null);
```

# 윈도우 이름과 윈도우 열기

51

## □ 이름 없는 윈도우 열기

```
<button onclick="window.open('http://www.naver.com', '',  
                                'width=600,height=600')">새 윈도우 열기  
</button>
```

- 버튼을 클릭할 때마다 새 윈도우를 열고 네이버 사이트 출력

## □ 이름을 가진 윈도우 열기

```
<button onclick="window.open('http://www.naver.com', 'myWin',  
                                'width=600,height=600')">새 윈도우 열기  
</button>
```

- myWin 이름의 윈도우가 열려 있지 않는 경우
  - 버튼을 클릭하면, myWin이름의 새 윈도우 열고 네이버 출력
- myWin 이름의 윈도우가 이미 열려 있는 경우
  - 버튼을 클릭하면, 이미 열려있는 myWin이름의 윈도우에 네이버 출력

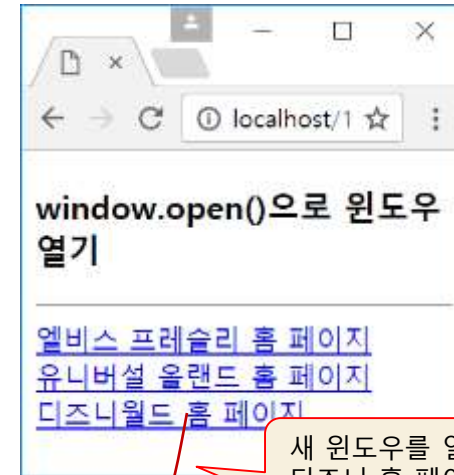
# 예제 10-1 window.open()으로 윈도우 열기

52

3개의 링크를 가진 웹 페이지를 작성하고, 각 링크를 클릭하면 myWin 이름의 새 윈도우를 열고 해당 사이트를 출력하라. myWin 윈도우는 공유된다. 새 윈도우는 스크린의 (300, 300) 위치에 800x700 크기로 출력된다.

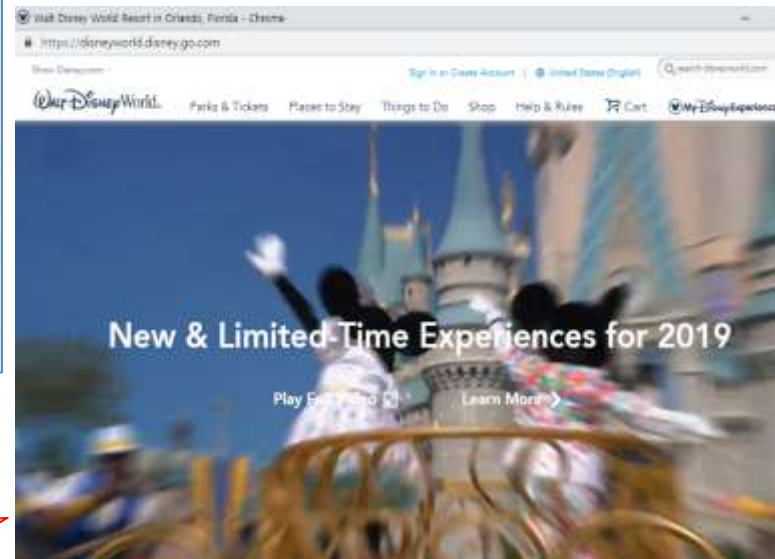
```
<!DOCTYPE html>
<html>
<head>
<title>윈도우 열기</title>
<script>
function load(URL) {
    window.open(URL, "myWin", "left=300,top=300,width=800,height=700");
}
</script>
</head>
<body>
<h3>window.open()으로 윈도우 열기</h3>
<hr>
<a href="javascript:load('http://www.graceland.com')">
    엘비스 프레슬리 홈 페이지</a><br>
<a href="javascript:load('http://www.universalorlando.com')">
    유니버설 올랜드 홈 페이지</a><br>
<a href="javascript:load('http://www.disneyworld.com')">
    디즈니월드 홈 페이지</a><br>
</body>
</html>
```

wk12js\_ex01\_BOM\_window\_open.html



새 윈도우를 열고  
디즈니 홈 페이지 출력

myWin 윈도우

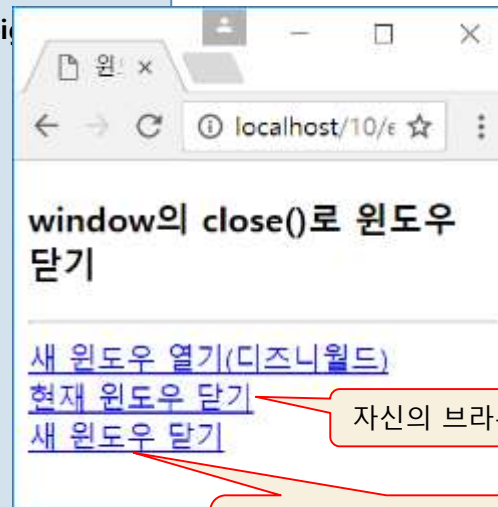


# 예제 10-2 윈도우 닫기

53

윈도우를 스스로 닫는 경우와 자신이 생성한 윈도우를 닫는 사례를 보인다.

```
<!DOCTYPE html>
<html>
<head>
<title>윈도우 닫기</title>
<script>
var newWin=null; // 새로 연 윈도우 기억
function load(URL) {
    newWin = window.open(URL, "myWin", "left=300,top=300,width=400,height=300");
}
function closeNewWindow() {
    if(newWin==null || newWin.closed) // 윈도우가 열리지 않았거나 닫힌 경우
        return; // 윈도우가 없는 경우 그냥 리턴
    else
        newWin.close(); // 열어 놓은 윈도우 닫기
}
</script>
</head>
<body>
<h3>window의 close()로 윈도우 닫기</h3>
<hr>
<a href="javascript:load('http://www.disneyworld.com')">
    새 윈도우 열기(디즈니월드)</a><br>
<a href="javascript>window.close()>
    현재 윈도우 닫기</a><br>
<a href="javascript:closeNewWindow()>
    새 윈도우 닫기</a>
</body>
</html>
```



wk12js\_ex02\_BOM\_window\_close.html

# window 객체의 타이머 활용

# window 객체의 타이머 활용

55

- window 객체의 타이머 기능 2 가지
  - ▣ 타임아웃 코드 1회 호출
    - `setTimeout() / clearTimeout()` 메소드
  - ▣ 타임아웃 코드 반복 호출
    - `setInterval() / clearInterval()` 메소드
- Mobile 환경에서 다양한 **simulation**에 사용

# setTimeout()/clearTimeout()

56

## □ setTimeout() : 타임아웃 코드 1회 실행

```
var timerID = setTimeout("timeOutCode", msec)
clearTimeout(timerID)
```

- timeOutCode : 타임아웃 자바스크립트 코드
- msec : 밀리초 단위의 정수로서, 타임아웃 지연 시간

setTimeout()은 msec 후에 timeOutCode를 1회 실행하도록 타이머를 설정하고, 타이머 ID를 리턴한다.  
clearTimeout()은 작동 중인 timerID의 타이머를 해제한다.

예) 3초 후 경고창 출력

```
function myAlert(time) {
    alert(time + "초 지났습니다");
}
var timerID = setTimeout("myAlert('3 sec')", 3000); // 3초 후 myAlert('3') 호출
```

예) 3초가 되기 전에 타이머 해제

```
clearTimeout(timerID); // timerID의 타이머 해제
```



# 예제 10-3 setTimeout()로 웹 페이지 자동 연결

57

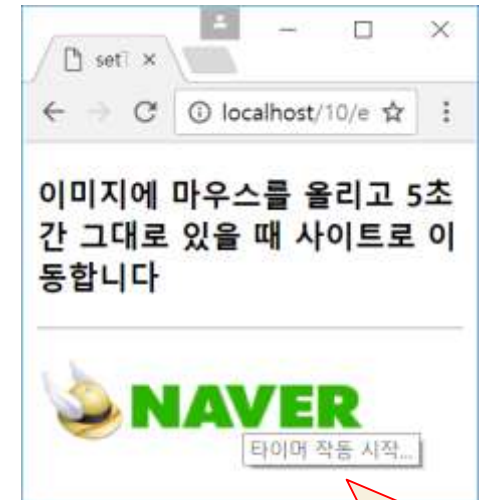
이미지 위에 마우스를 올린 상태로 5초가 지나면 네이버에 연결하며, 5초 전에 이미지를 벗어나면 타이머를 해제하는 코드를 작성하라.

```
<!DOCTYPE html>
<html>
<head>
<title>setTimeout()으로 웹 페이지 자동 연결</title>
</head>
<body>
<h3>이미지에 마우스를 올리고 5초간 그대로 있을 때 사이트로 이동합니다</h3>
<hr>

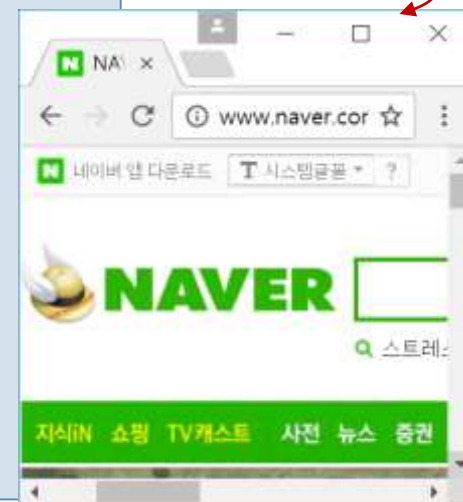
<script>
var timerID=null;
function startTimer(time) {
  // 타이머 시작
  timerID = setTimeout("load('http://www.naver.com')", time);

  // 이미지에 마우스 올리면 나타내는 툴팁 메시지
  document.getElementById("img").title = "타이머 작동 시작...";
}
function cancelTimer() {
  if(timerID !=null)
    clearTimeout(timerID); // 타이머 중단
}
function load(url) {
  window.location = url; // 현재 윈도우에 url 사이트 로드
}
</script>
</body>
</html>
```

wk12js\_ex03\_BOM\_set\_timeout.html



툴팁 메시지



마우스를 올리고  
5초간 그대로 있을 때

# setInterval() / clearInterval()

58

## □ setInterval() : 타임아웃 코드 반복 실행

```
var timerID = setInterval("timeOutCode", msec)
clearInterval(timerID)
```

- timeOutCode : 타임아웃 자바스크립트 코드
- msec : 밀리초 단위의 정수로서, 타임아웃 지연 시간

setInterval()은 msec 주기로 timeOutCode를 무한 반복하도록 타이머를 설정하고, 타이머의 ID를 리턴한다. clearInterval()은 timerID의 타이머를 해제한다.

예) 1초 간격으로 f() 반복 호출

```
function f() {
    // 함수 코드
}
var timerID = setInterval("f()", 1000); // 1초 주기로 f()가 호출되도록 타이머 작동
```

예) 타이머 해제

```
clearInterval(timerID); // timerID의 타이머 해제
```

# 예제 10-4 setInterval()로 텍스트 회전 (exam)

59

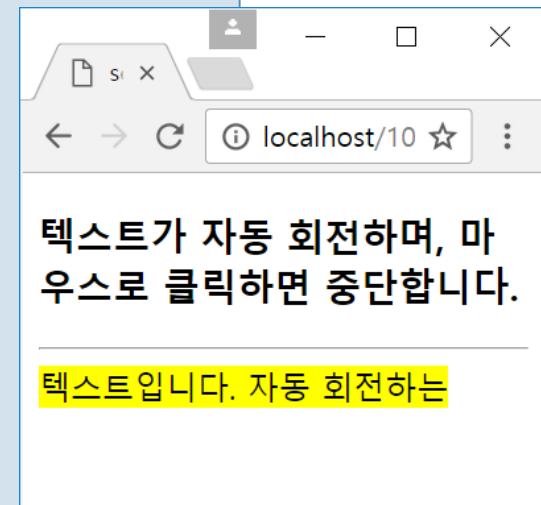
setInterval()을 이용하여 텍스트를 옆으로 반복 회전시키는 코드를 작성하라. 텍스트 위에 마우스를 클릭하면 회전이 중단된다.

```
<!DOCTYPE html>
<html>
<head> <title>setInterval()로 텍스트 회전</title> </head>
<body>
<h3>텍스트가 자동 회전하며, 마우스로 클릭하면 중단합니다.</h3>
<hr>
<div> <span id="div" style="background-color:yellow">
    자동 회전하는 텍스트입니다.</span>
</div>
<script>
var div = document.getElementById("div");
var timerID = setInterval("doRotate()", 200); // 200밀리초 주기로 doRotate() 호출

div.onclick = function (e) { // 마우스 클릭 이벤트 리스너
    clearInterval(timerID); // 타이머 해제. 문자열 회전 중단
}

function doRotate() {
    var str = div.innerHTML;
    var firstChar = str.substr(0, 1);
    var remains = str.substr(1, str.length-1);
    str = remains + firstChar;
    div.innerHTML = str;
}
</script>
</body> </html>
```

wk12js\_ex04\_BOM\_set\_inteval.html



# 윈도우 위치 및 크기 조절 (skip)

60

- 윈도우를 위로 5픽셀, 오른쪽으로 10픽셀 이동

```
window.moveBy(5, 10); 혹은  
moveBy(5, 10);
```

- 윈도우를 스크린의 (25, 10) 위치로 이동

```
window.moveTo(25, 10); 혹은 self.moveTo(25, 10);
```

- 윈도우 크기를 5 픽셀 좁게, 10픽셀 길게 조절

```
window.resizeBy(-5, 10); 혹은  
resizeTo(self.outerWifth-5, self.outerHeight+10);
```

- 윈도우 크기를 200x300으로 조절

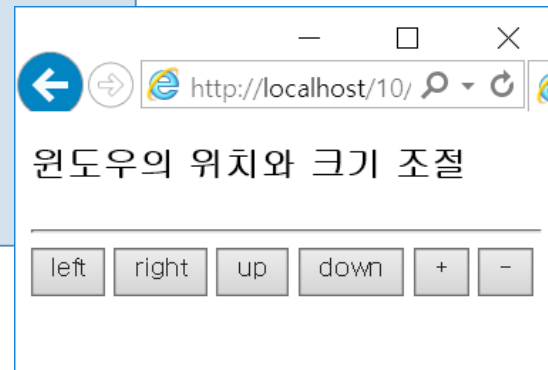
```
window.resizeTo(200, 300);
```

# 예제 10-5 윈도우의 위치와 크기 조절 (skip)

61

```
<!DOCTYPE html>
<html>
<head> <title> 윈도우의 위치와 크기 조절 </title> </head>
<body>
<h3> 윈도우의 위치와 크기 조절 </h3>
<hr>
<button onclick="window.moveBy(-10, 0)">left</button>
<button onclick="window.moveBy(10, 0)">right</button>
<button onclick="self.moveBy(0, -10)">up</button>
<button onclick="moveBy(0, 10)">down</button>
<button onclick="resizeBy(10, 10)">+</button>
<button onclick="resizeBy(-10, -10)">-</button>
</body>
</html>
```

wk12js\_ex05\_BOM\_window\_resize.html



\* 이 예제는 익스플로러에서는 잘 실행되지만, **Chrome**에서는 보안의 이유로 전혀 실행되지 않고, Edge에서는 크기 조절만 가능하다.

# 웹 페이지 스크롤

62

- 웹 페이지를 위로 10픽셀 스크롤(마우스 스크롤 다운)

```
window.scrollBy(0, 10); // 옆으로 0, 위로 10픽셀
```

- 웹 페이지를 왼쪽으로 10픽셀, 아래로 15픽셀 스크롤(마우스 스크롤 업)

```
window.scrollBy(10, -15);
```

- 웹 페이지의 (0, 200) 좌표 부분이 현재 윈도우의 왼쪽 상단 모서리에 출력되도록 스크롤

```
window.scrollTo(0, 200);
```

\* 스크롤 다운(scroll down)은 스크롤 바를 내리는 작동이며, 이에 따라 웹 페이지는 위로 이동한다.

# 예제 10-6 1초마다 10픽셀씩 자동 스크롤

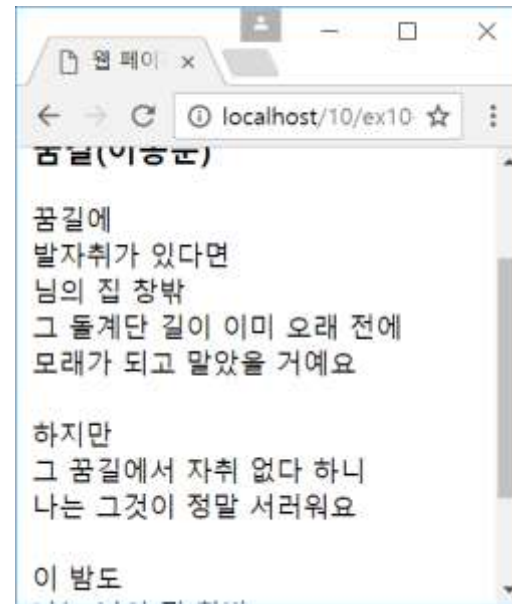
63

웹 페이지가 로드되자마자 자동으로 1초에 10픽셀씩 웹 페이지가 올라가도록 작성하라.

```
<!DOCTYPE html>
<html>
<head>
<title>웹 페이지의 자동 스크롤</title>
<script>
function startScroll(interval) {
  setInterval("autoScroll()", interval);
}

function autoScroll() {
  window.scrollBy(0,10); // 10픽셀 위로 이동
}
</script>
</head>
<body onload="startScroll(1000)">
<h3>자동 스크롤 페이지</h3>
<hr>
<h3>꿈길(이동순)</h3>
꿈길에<br>
발자취가 있다면<br>
님의 집 창밖<br>
그 돌계단 길이 이미 오래 전에<br>
모래가 되고 말았을 거예요<br><br>
하지만<br>
그 꿈길에서 자취 없다 하니<br>
나는 그것이 정말 서러워요<br><br>
이 밤도
```

```
그 꿈길에서 자취 없다 하니<br>
나는 그것이 정말 서러워요<br><br>
이 밤도<br>
나는 님의 집 창밖<br>
그 돌계단 위에 홀로 서서<br>
혹시라도 님의 목소리가 들려올까<br>
고개 숙이고 엿들어요<br>
</body>
</html>
```



wk12js\_ex06\_BOM\_window\_scroll.html

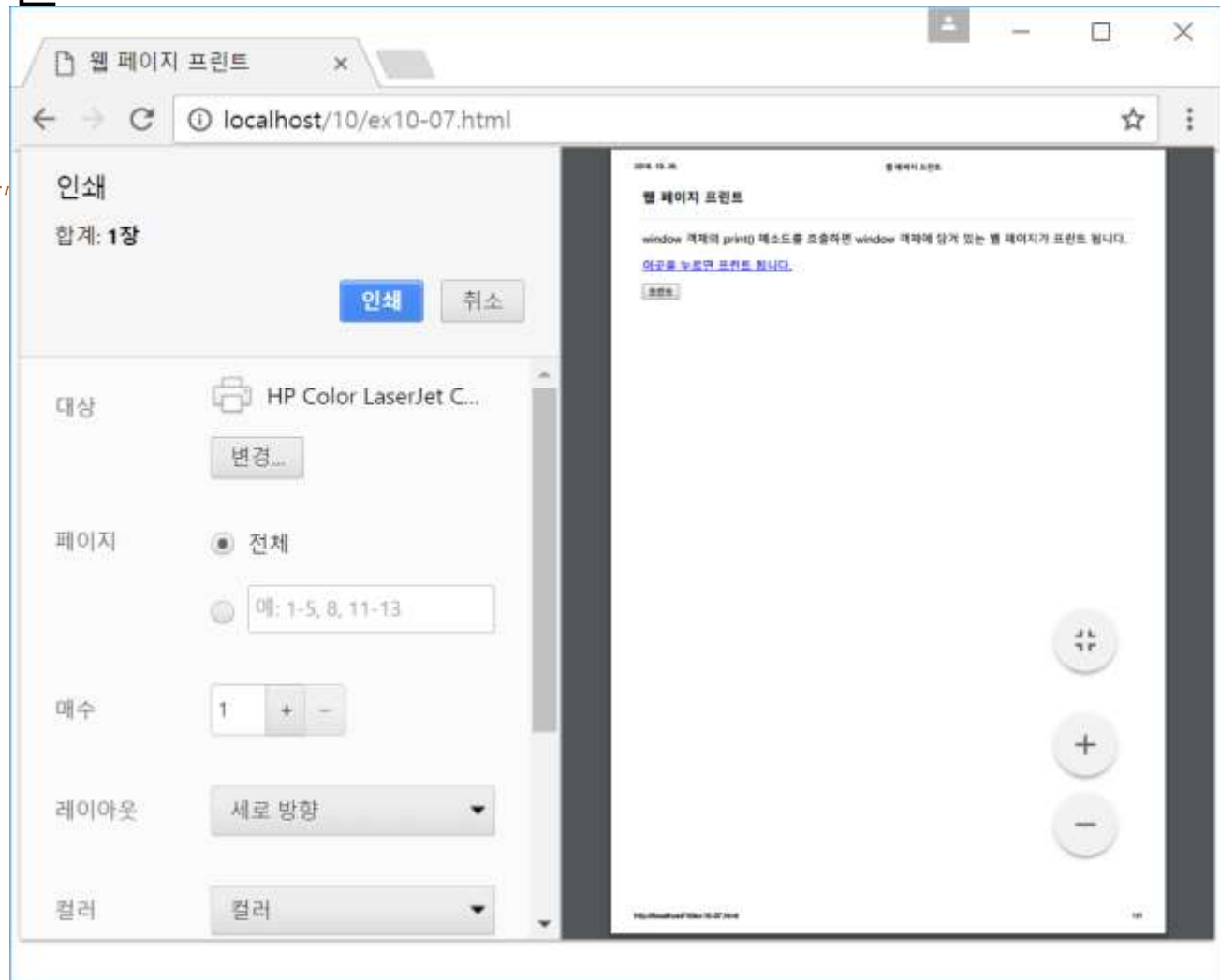
# 웹 페이지 프린트

64

## □ 웹 페이지 프린트

**window.print();**

이 코드가 실행되면  
인쇄 다이얼로그가 열리고,  
'확인' 버튼을 누르면  
인쇄가 이루어진다.



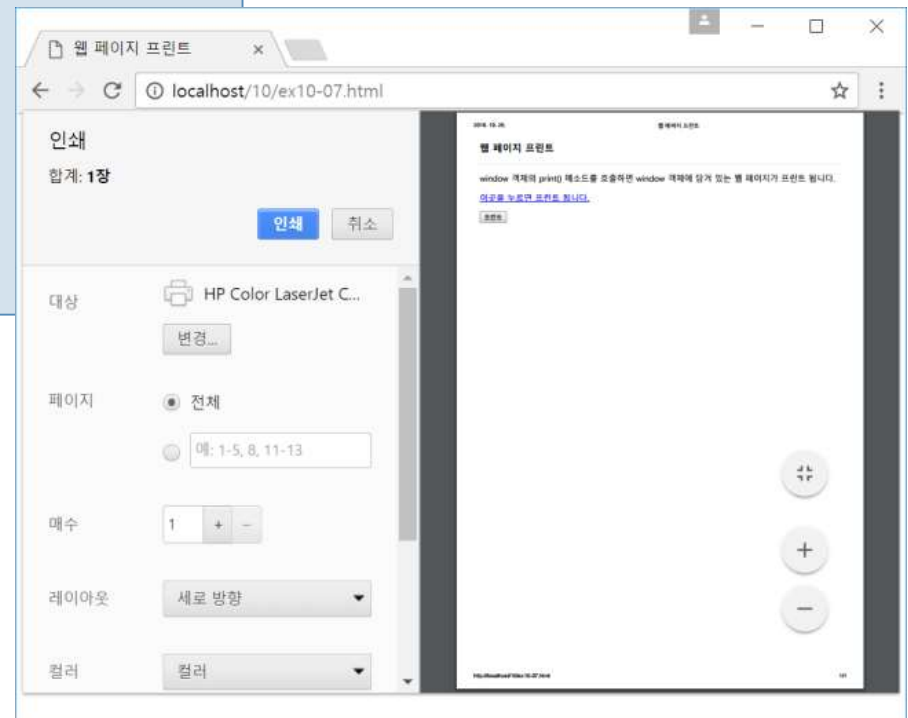
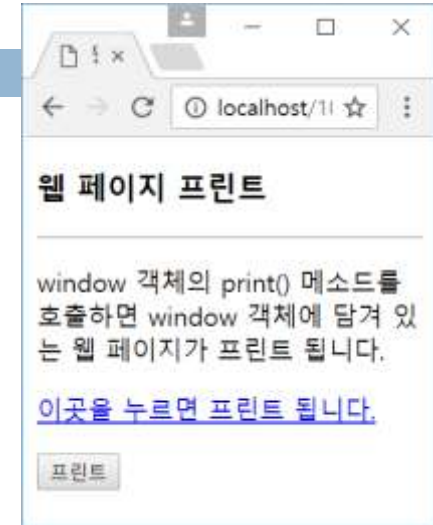


# 예제 10-7 웹 페이지 프린트

65

```
<!DOCTYPE html>
<html>
<head>
<title>웹 페이지 프린트</title> </head>
<body>
<h3>웹 페이지 프린트</h3>
<hr>
<p>window 객체의 print() 메소드를 호출하면
window 객체에 담겨 있는 웹 페이지가 프린트 됩니다.
<p>
<a href="javascript:window.print()">
  이곳을 누르면 프린트 됩니다.</a> <p>
<input type="button" value="프린트"
  onclick="window.print()">
</body>
</html>
```

wk12js\_ex07\_BOM\_page\_print.html



# onbeforeprint와 onafterprint

66

## □ 웹 페이지의 프린트 과정

1. window 객체에 onbeforeprint 리스너 호출
2. 웹 페이지 프린트

- 브라우저가 웹 페이지를 이미지로 만들어 프린터로 전송

3. window 객체에 onafterprint 리스너 호출

## □ onbeforeprint와 onafterprint 활용

- ▣ 웹 페이지에는 보이지 않는 회사 로고를 프린트 시 종이에 출력

### ▣ onbeforeprint

- 회사 로그 이미지를 보이도록 CSS3 스타일 설정

### ▣ onafterprint

- 회사 로그 이미지를 보이지 않도록 CSS3 스타일 설정

# 예제 10-8 onbeforeprint와 onafterprint 이벤트 활용

67

```
<!DOCTYPE html>
<html>
<head>
  <title>onbeforeprint와 onafterprint</title>
</head>
<body>
  <div id="logoDiv">
    
  </div>
  <p>안녕하세요. 브라우저 윈도우에서는 보이지 않지만, 프린트시에는 회사 로고가 출력되는 예제를 보입니다. 마우스 오른쪽 버튼을 눌러 인쇄 미리보기 메뉴를 선택해 보세요.</p>
</body>
</html>
```

67

```

<script>
  window.onbeforeprint=function (e) {
    logoDiv = document.getElementById("logoDiv");
    logoDiv.style.display = "block"; // block으로 변경. 로고가 화면에 나타나게 함
  }
  window.onafterprint=hideLogo;
  function hideLogo() {
    logoDiv = document.getElementById("logoDiv");
    logoDiv.style.display = "none"; // <div> 영역이 보이지 않게 함
    logoDiv.style.zIndex = -1; // 이미지를 문서의 맨 바닥으로 배치
  }
</script>
</head>
<body>
  <h3>onbeforeprint, onafterprint 이벤트 예</h3>
  <div id="logoDiv">
    
  </div>
  <p>안녕하세요. 브라우저 윈도우에서는 보이지 않지만, 프린트시에는 회사 로고가 출력되는 예제를 보입니다. 마우스 오른쪽 버튼을 눌러 인쇄 미리보기 메뉴를 선택해 보세요.</p>
</body>
</html>
```

wk12js\_ex08\_BOM\_onprint.html



이 예제는 익스플로러와 Edge에서는 실행되지만, Chrome에서는 실행되지 않는다. -> ?

# BOM

# location 객체

# location 객체

69

## □ location 객체

▣ 윈도우에 로드된 웹 페이지의 URL 정보를 나타내는 객체

▣ location 객체로 현재 윈도우에 웹 페이지 열기

```
window.location = "http://www.naver.com";  
window.location.href = "http://www.naver.com";  
window.location.assign("http://www.naver.com");  
window.location.replace("http://www.naver.com");
```

▣ 새 윈도우에 웹 페이지 열기

```
var win=window.open();           // 빈 윈도우 열기  
win.location="http://www.naver.com"; // 네이버 페이지 로드
```

▣ location 객체의 프로퍼티와 URL의 구성 요소와의 관계

href

protocol	hostname	port	pathname	hash
http://	www.mysite.com	:8080	/content/URL분석.html	#label1

host

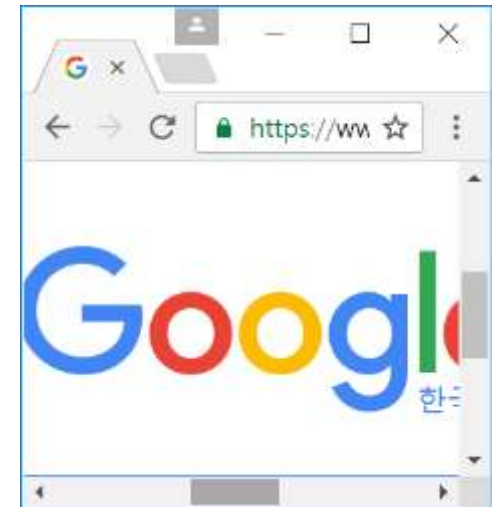
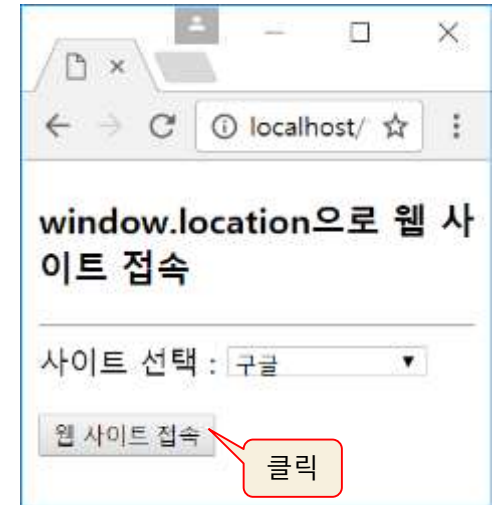
http://search.naver.com/search.naver?query=javascript

pathname	search
/search.naver	?query=javascript

# 예제 10-9 location 객체로 웹 사이트 접속 [DIY]

70

```
<!DOCTYPE html>
<html>
<head> <title>window.location으로 웹 사이트 접속</title>
<script>
function load() {
    var select = document.getElementById("site");
    window.location=select.options[select.selectedIndex].value;
}
</script>
</head>
<body>
<h3>window.location으로 웹 사이트 접속</h3>
<hr>
사이트 선택 :
<select id="site">
    <option value="http://www.naver.com" selected> 네이버
    <option value="http://www.google.com"> 구글
    <option value="http://www.microsoft.com"> 마이크로소프트
</select>
<p>
<button onclick="load()"> 웹 사이트 접속</button>
</body>
</html>
```



# HTML5 Geolocation

[< Previous](#)[Next >](#)

The HTML Geolocation API is used to locate a user's position.

[Try It](#)

[https://www.w3schools.com/html/html5\\_geolocation.asp](https://www.w3schools.com/html/html5_geolocation.asp)

# BOM

# navigator 객체



# navigator 객체

73

## □ navigator 객체

- ▣ 현재 작동 중인 브라우저에 대한 다양한 정보를 나타내는 객체

프로퍼티	설명	r/w
appName	브라우저의 코드 이름을 가진 문자열	r
appVersion	브라우저 이름 문자열	r
platform	브라우저의 플랫폼과 버전에 관한 문자열	r
product	운영체제 플랫폼의 이름	r
userAgent	브라우저 엔진의 이름	r
vendor	브라우저가 웹 서버로 데이터를 전송할 때, HTTP 헤더 속의 user-agent 필드에 저장하는 문자열로서 웹 서버가 클라이언트를 인식하기 위한 목적	r
language	브라우저 제작 회사의 이름 문자열	r
onLine	브라우저의 언어를 나타내는 문자열로서, 영어는 "en-US", "ko-KR"	r
plugins	브라우저가 현재 온라인 작동중이면 true, 아니면 false	r
cookieEnabled	브라우저에 설치된 플러그인(plugin 객체)에 대한 컬렉션	r
geolocation	브라우저에 쿠키를 사용할 수 있는 상태이면 true, 아니면 false	r
	위치 정보를 제공하는 geolocation 객체에 대한 레퍼런스	r

# 예제 10-10 navigator로 브라우저 정보 출력

74

```
<!DOCTYPE html>
<html>
<head><title>브라우저 정보 출력</title>
<style>
span { color : red; }
div {
    border-color : yellowgreen;
    border-style : solid;
    padding : 5px;
}
</style>
<script>
function printNavigator() {
    var text = "<span>appName</span>: " + navigator.appCodeName + "<br>";
    text += "<span>appName</span>: " + navigator.appName + "<br>";
    text += "<span>appVersion</span>: " + navigator.appVersion + "<br>";
    text += "<span>platform</span>: " + navigator.platform + "<br>";
    text += "<span>product</span>: " + navigator.product + "<br>";
    text += "<span>userAgent</span>: " + navigator.userAgent + "<br>";
    text += "<span>vendor</span>: " + navigator.vendor + "<br>";
    text += "<span>language</span>: " + navigator.language + "<br>";
    text += "<span>onLine</span>: " + navigator.onLine + "<br>";
    text += "<span>cookieEnabled</span>: " + navigator.cookieEnabled + "<br>";
    text += "<span>javaEnabled</span>: " + navigator.javaEnabled() + "<br>";
    text += "<span>plugins.length</span>: " + navigator.plugins.length + "<br>";
    for(j=0; j<navigator.plugins.length; j++) {
        text += "plugins" + j + " : <blockquote>";
        text += navigator.plugins[j].name + "<br>";
        text += "<i>" + navigator.plugins[j].description + "</i> <br>";
        text += navigator.plugins[j].filename + "</blockquote>";
    }

    // div 태그에 출력
    var div = document.getElementById("div");
    div.innerHTML = text;
}
```

```
</script>
</head>
<body onload="printNavigator()">
<h3>브라우저에 관한 정보 출력</h3>
아래에 이 브라우저에 관한 여러 정보를 출력합니다.
<hr>
<p>
<div id="div"></div>
</body>
</html>
```



플러그인 이름

플러그인 설명

플러그인 파일

# BOM

# screen 객체

# screen 객체

77

## □ screen

- ▣ 브라우저가 실행되는 스크린 장치에 관한 정보를 담고 있는 객체

프로퍼티	설명	r/w
availHeight	작업 표시줄 등을 제외하고 브라우저가 출력 가능한 영역의 높이	r
availWidth	작업 표시줄 등을 제외하고 브라우저가 출력 가능한 영역의 폭	r
pixelDepth	한 픽셀의 색을 나타내기 위해 사용되는 비트 수	r
colorDepth	한 픽셀의 색을 나타내기 위해 사용되는 비트 수로서 pixelDepth와 동일. 대부분의 브라우저에서 지원되므로 pixelDepth보다 colorDepth를 사용할 것을 권장	r
height	스크린의 수직 픽셀 수	r
width	스크린의 수평 픽셀 수	r

# 예제 10-11 스크린 장치에 관한 정보 출력

78

wk12js\_ex11\_BOM\_screen.html

```
<!DOCTYPE html>
<html>
<head>
<title>스크린 장치에 관한 정보 출력</title>
<script>
function printScreen() {
    var text = "availHeight:".fontcolor('blue') + screen.availHeight + "<br>";
    text += "availWidth:".fontcolor('blue') + screen.availWidth + "<br>";
    text += "colorDepth:".fontcolor('blue') + screen.colorDepth + "<br>";
    text += "pixelDepth:".fontcolor('blue') + screen.pixelDepth + "<br>";
    text += "height:".fontcolor('blue') + screen.height + "<br>";
    text += "width:".fontcolor('blue') + screen.width + "<br>";

    document.getElementById("div").innerHTML = text;
}
</script>
</head>
<body onload="printScreen()">
<h3>스크린 장치에 관한 정보</h3>
<hr>
<div id="div"></div>
</body>
</html>
```

## 스크린 장치에 관한 정보

availHeight:1890

작업 표시줄 높이 제외

availWidth:1080

colorDepth:24

픽셀당 2<sup>24</sup> 색

pixelDepth:24

height:1920

스크린 크기

width:1080

height와 width는 브라우저의 설정에서 확대/축소 값을 100%로 해야 정확한 값으로 출력됨

# BOM

# history 객체

# history 객체

80

## □ history 객체

- 윈도우에서 방문한 웹 페이지 리스트(히스토리)를 나타내는 객체

프로퍼티	설명	r/w
length	히스토리 리스트에 있는 엔트리 수	r

메소드	설명
back()	히스토리에 있는 이전 웹 페이지로 이동. 브라우저의 <back> 버튼과 동일
forward()	히스토리에 있는 다음 웹 페이지로 이동. 브라우저의 <forward> 버튼과 동일
go(n)	히스토리에서 현재 웹 페이지에서 n 만큼 상대적인 웹 페이지로 이동

- history 객체를 이용하여 웹 페이지를 이동하는 코드 사례

```
history.back();    // 이전 페이지로 이동
history.go(-1);    // 이전 페이지로 이동
history.forward(); // 다음 페이지로 이동
history.go(1);     // 다음 페이지로 이동
```

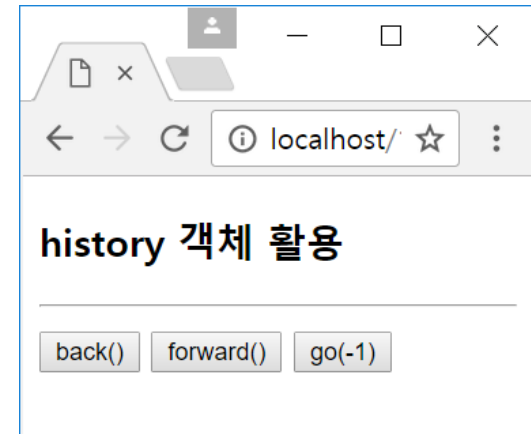


# 예제 10-12 history 객체 활용

81

```
<!DOCTYPE html>
<html>
<head> <title>history 객체 활용 </title> </head>
<body>
<h3>history 객체 활용</h3>
<hr>
<button onclick="history.back()">back()</button>
<button onclick="history.forward()">forward()</button>
<button onclick="history.go(-1)">go(-1)</button>
</body>
</html>
```

wk12js\_ex12\_BOM\_history.html



[DIY] 자바스크립트 DOM, BOM을 활용하는 두 개의 html 문서를 만드시오.

1. [예제 8-4 this 활용 → 예제 8-6 참고]  
아래에 “Go back!” 버튼을 추가하고 클릭하면  
모든 버튼을 초기 모양으로 되돌리는 html 문서를  
작성하시오.

파일명: [hmn\\_this.html](#)

this 활용

버튼

버튼

여기 클릭하면 크기와 색 변경

Go back!

2. [예제 10-9 location 활용]

“hmn favorite 3 XX” 라는 제목으로 영화, 여행, 노래, 취미 중 하나의 주제를 선정 하시오.  
선택된 주제에 대한 Best 3 정보를 select tag에 담아 그 중 하나를 선택해서 보여주는 html  
문서를 작성하시오.

파일명: [hmn\\_best3.html](#)

가점: Javascript 프로그래밍 응용 능력.

# Possible result ...

83

## this 활용



여기 클릭하면 크기와 색 변경

Go back!




## this 활용



여기 클릭하면 크기와 색 변경

Go back!

```
<script>
  function change(obj, size, color) {
    obj.style.color = color;
    obj.style.fontSize = size;
  }

  function reset() {
    var resetArray = document.getElementsByClassName("button");
    
  }
</script>
```

```
<button class="button" onclick="change(this, '30px', 'red')">
  버튼
</button>
<button class="button" onclick="change(this, '30px', 'blue')">
  버튼
</button>
<div class="button" onclick="change(this, '25px', 'orange')">
  여기 클릭하면 크기와 색 변경
</div>
<button onclick="reset()">
  go back
</button>
```

# Good report

84

## this 활용하여 과제 해결



여기 클릭하면 크기와 색 변경

GoBack\_btn!!



## this 활용하여 과제 해결



여기 클릭하면 크기와 색 변경

GoBack\_btn!!

```
<script>
function change(obj, size, color) {
    obj.style.color = color;
    obj.style.fontSize = size;
}

function reset() {
    var resetArray = document.getElementsByClassName("button");
    for (var i = 0; i < resetArray.length; i++) {
        var reset = resetArray[i];
        reset.style.color = null;
        reset.style.fontSize = null;
    }
}
</script>
```

```
<button class="button" onclick="change(this, '30px', 'red')">
    버튼
</button>
<button class="button" onclick="change(this, '30px', 'blue')">
    버튼
</button>
<div class="button" onclick="change(this, '25px', 'orange')">
    여기 클릭하면 크기와 색 변경
</div>
<button onclick="reset()">
    go back
</button>
```

# wk12-실습 : 결과를 나의 github에 올리기

85

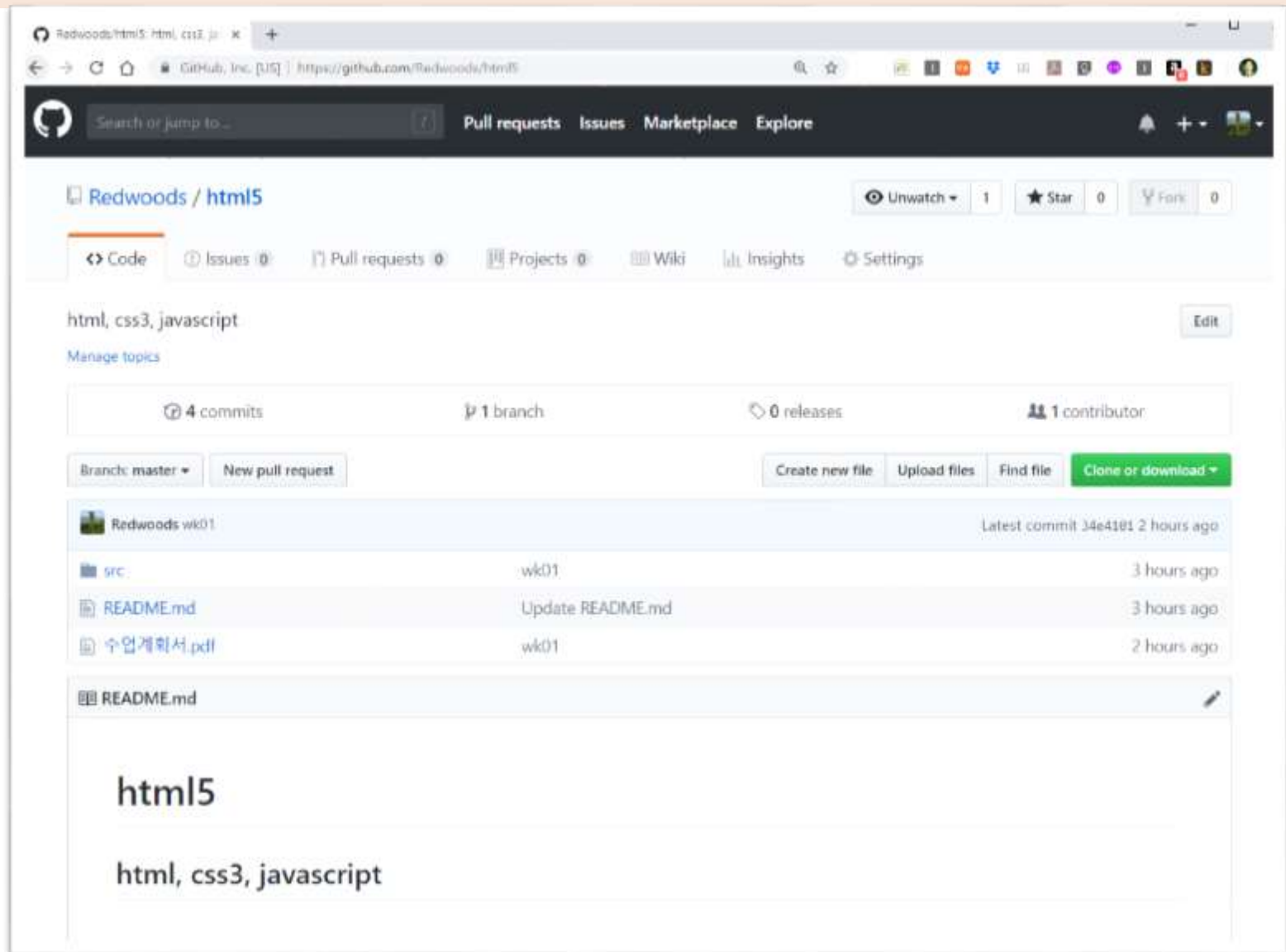
실습 결과를 github에 올립니다.

1. README.md에는 실습 결과 요약 추가 입력
2. hmxx\_this.html 완성
3. hmxx\_best3.html 완성
4. "hmxx" repo의 wk12 폴더에 upload

단 업로드가 안될 경우, wk12.zip을 업로드  
그리고 집에서 wk12 폴더로 다시 업로드.

# HTML5 : 강의자료실

## <https://github.com/redwoods/html5>



# 교재 WEB 강의 소개

← → ↻ ① webprogramming.co.kr ☆

명품 HTML5+ CSS3+ Javascript 웹 프로그래밍

Home Introduction Notice Board Support Code

명품 HTML5 + CSS3 + Javascript 웹 프로그래밍

HTML5로 여러분의 무한한 상상력을 표현해 보세요!



Sir Tim Berners-Lee (1955.6.8 ~)

명품 웹 프로그래밍 소개  
"웹 프로그래밍을 가장 쉽게 익힐 수 있는 책"

처음 웹 프로그래밍을 공부하는 입문자들도 모든 주제를 직관적으로 이해하고 빠르게 파악할 수 있습니다.



자세히보기 →

강력한 Q&A 피드백 제공  
"빠르고, 간결하고, 정확한 저자의 직접적인 답변"

'이거 이해가 잘 안되는데... 물어볼 사람도 없고...'  
더이상 고민하지 마세요.  
명품 웹 프로그래밍 홈페이지에서는 누구나 저자가 직접 답변해주는 Q&A 게시판들을 이용할 수 있습니다.



자세히보기 →

즉석 실행 가능한 예제 프로그램  
"백문이 불여일견, 백견이 불여일타(打)!"

코드뿐만 실행되어 있는 예제들, 결과 화면이 있어도 이해가 잘 안되시죠?  
예제 소스를 바탕으로, 내맘대로 수정한 코드를 즉석으로 웹 페이지로 변환해주는 예제 프로그램을 통해 모든 코드를 빠르고 쉽게 이해할 수 있습니다.



자세히보기 →

Notice

Test

2017-01-16 15:32

Know-How

Test

2017-01-17 14:04 관리자

87

<http://webprogramming.co.kr>

# 관련 WEB 강의 소개 – w3schools.com

The screenshot shows the w3schools.com website. The browser address bar displays "https://www.w3schools.com". The website has a green header with the logo "w3schools.com" and the tagline "THE WORLD'S LARGEST WEB DEVELOPER SITE". Below the header is a navigation bar with "TUTORIALS", "REFERENCES", and "EXAMPLES". On the left side, there is a sidebar menu with links to various topics: HTML and CSS, JavaScript, Server Side, Web Building, and XML Tutorials. The main content area is divided into three sections: HTML, CSS, and JavaScript. Each section has a title, a subtitle, and a "Try It Yourself" button. The HTML section includes a code example for a basic HTML document. The CSS section includes a code example for styling a body and a heading. The JavaScript section includes a code example for a function that gets an element by ID and changes its style.

HTML and CSS

- Learn HTML
- Learn CSS
- Learn W3.CSS
- Learn Colors
- Learn Bootstrap
- Learn Icons
- Learn Graphics
- Learn How To

JavaScript

- Learn JavaScript
- Learn W3.JS
- Learn JQuery
- Learn JQueryMobile
- Learn AppML
- Learn AngularJS
- Learn JSON
- Learn AJAX

Server Side

- Learn SQL
- Learn PHP
- Learn ASP

Web Building

- Web Templates
- Web Statistics
- Web Certificates

XML Tutorials

- Learn XML
- Learn XML AJAX
- Learn XML DOM
- Learn XML DTD
- Learn XML Schema
- Learn XSLT
- Learn XPath
- Learn XQuery

## HTML

The language for building web pages

LEARN HTML HTML REFERENCE

HTML Example:

```
<!DOCTYPE html>
<html>
<title>HTML Tutorial</title>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Try It Yourself »

## CSS

The language for styling web pages

LEARN CSS CSS REFERENCE

CSS Example:

```
body {
  background-color: lightblue;
}
h1 {
  color: white;
  text-align: center;
}
p {
  font-family: verdana;
  font-size: 28px;
}
```

Try It Yourself »

## JavaScript

The language for programming web pages

JavaScript Example:

```
<script>
function myFunction() {
  var x = document.getElementById("demo");
  x.style.backgroundColor = "red";
}
```